



ΠΑΝΕΠΙΣΤΗΜΙΟ ΔΥΤΙΚΗΣ ΑΤΤΙΚΗΣ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΗΣ ΣΧΕΔΙΑΣΗΣ ΚΑΙ ΠΑΡΑΓΩΓΗΣ

ΘΕΜΑ ΔΙΠΛΩΜΑΤΙΚΗΣ ΕΡΓΑΣΙΑΣ

"ΕΞΥΠΝΟ ΓΑΝΤΙ ΜΕ ΔΥΝΑΤΟΤΗΤΑ ΕΚΜΑΘΗΣΗΣ ΚΙΝΗΣΕΩΝ ΚΑΙ ΦΩΝΗΤΙΚΩΝ
ΕΝΤΟΛΩΝ ΚΑΘΩΣ ΚΑΙ ΑΠΟΣΤΟΛΗΣ ΔΕΔΟΜΕΝΩΝ ΜΕΣΩ BLUETOOTH"



ΟΝΟΜΑ ΦΟΙΤΗΤΗ:

ΚΩΤΣΑΙΝΑΣ ΠΑΝΑΓΙΩΤΗΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:

ΔΡΟΣΟΣ ΧΡΗΣΤΟΣ

ΑΙΓΑΛΕΩ, ΙΟΥΛΙΟΣ 2023

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο κάτωθι υπογεγραμμένος Παναγιώτης Κώτσαινας του Χαράλαμπου, με αριθμό μητρώου 71445294 φοιτητής του Πανεπιστημίου Δυτικής Αττικής της Σχολής Μηχανικών του Τμήματος Μηχανικών Βιομηχανικής Σχεδίασης & Παραγωγής, δηλώνω υπεύθυνα ότι:

«Είμαι συγγραφέας αυτής της πτυχιακής/διπλωματικής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης, οι όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε ακριβώς είτε παραφρασμένες, αναφέρονται στο σύνολό τους, με πλήρη αναφορά στους συγγραφείς, τον εκδοτικό οίκο ή το περιοδικό, συμπεριλαμβανομένων και των πηγών που ενδεχομένως χρησιμοποιήθηκαν από το διαδίκτυο. Επίσης, βεβαιώνω ότι αυτή η εργασία έχει συγγραφεί από μένα αποκλειστικά και αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο δικής μου, όσο και του Ιδρύματος.

Παράβαση της ανωτέρω ακαδημαϊκής μου ευθύνης αποτελεί ουσιώδη λόγο για την ανάκληση του πτυχίου μου».

Ο Δηλών



ΠΕΡΙΛΗΨΗ

Η παρούσα διπλωματική εργασία εστιάζει στη δημιουργία ενός έξυπνου γαντιού με δυνατότητες αναγνώρισης κινήσεων και φωνητικών εντολών, με σκοπό τη βελτίωση της επικοινωνίας και της αυτονομίας των ατόμων με οπτικές και κινητικές δυσκολίες. Για να επιτευχθεί ο σκοπός χρησιμοποιούνται διάφοροι αισθητήρες όπως επιταχυνσιόμετρο, μικρόφωνο, οπτικοί αισθητήρες και περιβαλλοντικοί αισθητήρες, καθώς και τεχνικές μηχανικής μάθησης (Machine Learning) και της τεχνητής νοημοσύνης (Artificial Intelligence) για εκμάθηση κινήσεων και φωνητικών εντολών. Το γάντι επικοινωνεί μέσω Bluetooth και προβάλλει τα δεδομένα σε μια οθόνη με γραφική απεικόνιση. Τα αποτελέσματα παρέχουν προηγμένες λειτουργίες για την κατανόηση των συνθηκών περιβάλλοντος και την επικοινωνία με το περιβάλλον. Μελλοντικές βελτιώσεις περιλαμβάνουν την ανάπτυξη περισσότερων εφαρμογών και λειτουργιών για ευκολότερη προσαρμογή στις ανάγκες των χρηστών με οπτικές και κινητικές δυσκολίες.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Arduino, Τεχνητή Νοημοσύνη, Μηχανική μάθηση

ABSTRACT

The purpose of this thesis is to develop a smart glove with motion recognition and voice command capabilities. These functionalities will be achieved using sensors such as accelerometers, microphones, light sensors, color recognition sensors, and temperature sensors. The main tools employed for this research will be Machine Learning and Artificial Intelligence, used to learn and interpret hand gestures and voice commands. The application of this smart glove primarily targets individuals with mobility difficulties. The glove will transmit data via Bluetooth and display it on a graphical interface. Additionally, the glove features LED indicators that allow users to adjust the color settings. It can be utilized for color recognition either through visual display or voice output. Users can train the glove with specific words and gestures to create shortcuts and adapt to new environmental conditions. Future enhancements may involve the development of additional applications and functionalities to better cater to the needs of individuals with visual-motor difficulties.

KEYWORDS

Arduino, Artificial Intelligence, Machine Learning

ΤΡΙΜΕΛΗΣ ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ

ΔΡΟΣΟΣ ΧΡΗΣΤΟΣ	
ΛΑΣΚΑΡΗΣ ΝΙΚΟΣ	
ΣΥΜΕΩΝΑΚΗ ΕΛΕΝΗ	

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ	ii
ΠΕΡΙΛΗΨΗ.....	iii
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ.....	iii
ABSTRACT	iv
KEYWORDS	iv
ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ	vi
ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ	viii
ΑΠΟΔΟΣΗ ΟΡΩΝ.....	ix
ΕΙΣΑΓΩΓΗ.....	1
ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ	2
ΟΡΙΣΜΟΣ ΠΡΟΒΛΗΜΑΤΟΣ.....	5
ΣΧΕΔΙΑΣΗ ΚΑΙ ΚΑΤΑΣΚΕΥΗ ΥΛΙΚΟΥ ΜΕΡΟΥΣ (HARDWARE)	6
Σενάρια Λειτουργίας	6
Λειτουργικά Μέρη Κατασκευής	8
Μηχανολογικό σχέδιο	16
Ηλεκτρονικό κύκλωμα	17
Τυπωμένο κύκλωμα.....	19
ΑΝΑΛΥΣΗ ΚΑΙ ΣΥΓΓΡΑΦΗ ΛΟΓΙΣΜΙΚΟΥ (SOFTWARE)	20
Διαγράμματα Ροής	20
• Βρόχος: Στο Βρόχο ελέγχεται η σύνδεση του γαντιού με συσκευή Bluetooth και διαβάζονται οι αισθητήρες.	20
• Συνάρτηση Αισθητήρων Δακτύλων: Διαβάζει την πίεση στα δάκτυλα.....	24
• Συνάρτηση Αισθητήρων Ήχου: Διαβάζει τα δεδομένα ήχου από τον πυκνωτή	25
• Συνάρτηση Διαβάσματος τιμών εισόδου από εφαρμογή: Αναγνωρίζει τις τιμές διάφορων χαρακτηριστικών που προέρχονται από την εφαρμογή.....	26
• Συνάρτηση λειτουργίας LED: ενημερώνει το χρώμα των φωτεινών δεικτών	27
Βασικός Κώδικας Εφαρμογής Android.....	28
Βασικός Κώδικας Arduino	34
Βιβλιοθήκες.....	34
Χαρακτηριστικά Bluetooth.....	34
Μεταβλητές	35

Set up.....	37
Βρόχος.....	40
Συνάρτηση Περιβαλλοντικών Αισθητήρων	41
Συνάρτηση Οπτικών Αισθητήρων.....	42
Συνάρτηση Αισθητήρα Κλίσης	43
Συνάρτηση Αισθητήρων Δακτύλων	45
Συνάρτηση Αισθητήρων Ήχου.....	46
Συνάρτηση Διαβάσματος τιμών εισόδου από εφαρμογή	47
Συνάρτηση λειτουργίας LED	53
ΣΥΜΠΕΡΑΣΜΑΤΑ.....	57
ΒΙΒΛΙΟΓΡΑΦΙΑ.....	58
ΠΑΡΑΡΤΗΜΑ	60

ΚΑΤΑΛΟΓΟΣ ΔΙΑΓΡΑΜΜΑΤΩΝ

Εικόνα 1 Λειτουργία επιταχυνσιόμετρου.....	3
Εικόνα 2 Επικοινωνία Γάντι - Εφαρμογή.....	7
Εικόνα 3 Λειτουργία κώδικα Arduino.....	7
Εικόνα 4 Αισθητήρας APDS9960.....	9
Εικόνα 5 Αισθητήρας LPS22HB.....	9
Εικόνα 6 Αισθητήρας HTS221.....	10
Εικόνα 7 Αισθητήρας LSM9DS1.....	10
Εικόνα 8 Διεπαφή χρήστη με διαδικτυακή εφαρμογή.....	12
Εικόνα 9 Εικόνες Εφαρμογής με περιγραφή.....	15
Εικόνα 10 Μηχανολογικό σχέδιο.....	16
Εικόνα 11 Έξοδοι Arduino.....	17
Εικόνα 12 Κυκλώματα φόρτισης, αλλαγή λογικής κατάστασης και σημεία σύνδεσης.....	17
Εικόνα 13 Κύκλωμα LED.....	18
Εικόνα 14 Τυπωμένο Κύκλωμα.....	19
Εικόνα 15 Διάγραμμα Ροής Loop.....	20
Εικόνα 16 Διάγραμμα Ροής Συνάρτησης Περιβαλλοντικών Αισθητήρων.....	21
Εικόνα 17 Διάγραμμα Ροής Συνάρτησης Οπτικών Αισθητήρων.....	22
Εικόνα 18 Διάγραμμα Ροής Συνάρτησης Αισθητήρα Κλίσης.....	23
Εικόνα 19 Διάγραμμα Ροής Συνάρτησης Αισθητήρων Δακτύλων.....	24
Εικόνα 20 Διάγραμμα Ροής Συνάρτησης Αισθητήρων Ήχου.....	25
Εικόνα 21 Διάγραμμα Ροής Συνάρτησης Διαβάσματος τιμών εισόδου από εφαρμογή.....	27
Εικόνα 22 Διάγραμμα Ροής Συνάρτησης λειτουργίας LED.....	27
Εικόνα 23 Αρχικοποίηση μεταβλητών (Αριστερά) Χαρακτηριστικά Bluetooth (Δεξιά).....	28
Εικόνα 24 Άδειες Bluetooth, κουμπί σύνδεσης και κουμπί αποσύνδεσης.....	28
Εικόνα 25 Ενεργοποίηση κουμπιών μετά τη σύνδεση.....	29
Εικόνα 26 Εγγραφή στα Χαρακτηριστικά Bluetooth.....	29
Εικόνα 27 Ενέργειες κατά τη διακοπή σύνδεσης.....	30
Εικόνα 28 Λειτουργίες όταν ληφθούν bytes από το γάντι.....	32
Εικόνα 29 Λειτουργίες όταν ληφθούν bytes από το γάντι.....	32
Εικόνα 30 Διαδικασία για κουμπίά εξαγώνου Πάνω-Κάτω.....	33
Εικόνα 31 Διακόπτες για τον έλεγχο των χρωμάτων.....	33
Εικόνα 32 Κουμπίά για ενεργοποίηση απενεργοποίηση όλων των Led.....	33

ΑΠΟΔΟΣΗ ΟΡΩΝ

- Μικροελεγκτής: «είναι ένας ολοκληρωμένος μικροϋπολογιστής, σχεδιασμένος πάνω σε ένα ολοκληρωμένο που εμπεριέχει, όχι μόνο μικροεπεξεργαστή αλλά και RAM, ROM, χρονιστές, UART, πόρτες και άλλες κοινές, περιφερειακές λειτουργίες εισόδου-εξόδου» (Πογαρίδης, 2015).
- System on Chip (SoC): Ένα System on Chip (SoC) αναφέρεται σε ένα μοναδικά ενσωματωμένο κύκλωμα (chip) που αποτελείται από όλα τα εξαρτήματα ενός ηλεκτρονικού συστήματος (Risset, 2011).

ΕΙΣΑΓΩΓΗ

Η παρούσα διπλωματική εργασία έχει ως σκοπό τη δημιουργία ενός έξυπνου γαντιού, γνωστού και ως "smart glove", που απευθύνεται κυρίως σε άτομα με οπτικές και κινητικές δυσκολίες, προσφέροντάς τους ένα εργαλείο που τους βοηθά στην επικοινωνία και την αλληλεπίδραση με το περιβάλλον τους. Το γάντι ενσωματώνει προηγμένες δυνατότητες αναγνώρισης κινήσεων και φωνητικών εντολών και είναι εξοπλισμένο με διάφορους αισθητήρες, όπως επιταχυνσιόμετρο, μικρόφωνο, αισθητήρα φωτεινότητας, αισθητήρα αναγνώρισης χρωμάτων και θερμοκρασίας. Τα δεδομένα που συλλέγονται από αυτούς τους αισθητήρες μεταδίδονται μέσω τεχνολογίας Bluetooth και προβάλλονται σε μια οθόνη με γραφική απεικόνιση.

Οι βασικές λειτουργίες του έξυπνου γαντιού περιλαμβάνουν την αναγνώριση κινήσεων του χεριού και την αναγνώριση φωνητικών εντολών. Για την υλοποίηση αυτών των λειτουργιών, χρησιμοποιούνται τεχνικές μηχανικής μάθησης (ML) και τεχνητής νοημοσύνης (AI), προκειμένου να γίνει η εκπαίδευση του συστήματος στην αναγνώριση και την ερμηνεία των κινήσεων και των φωνητικών εντολών.

Αξιοποιώντας τους ενσωματωμένους αισθητήρες του, το γάντι έχει τη δυνατότητα να ανιχνεύσει τις κινήσεις του χεριού, όπως η κίνηση προς τα πάνω ή κάτω, καθώς και την κίνηση του καρπού, επιτρέποντας στους χρήστες να εκτελούν εντολές με χειρονομίες. Επιπλέον, οι φωτεινές ενδείξεις LED που διαθέτει επιτρέπουν στους χρήστες να ρυθμίζουν τον χρωματισμό τους, είτε με τη χρήση του αισθητήρα αναγνώρισης χρώματος είτε με φωνητικές εντολές. Η χρήση του έξυπνου γαντιού προβλέπεται να δημιουργήσει νέες δυνατότητες στον τομέα της επικοινωνίας και της ανεξάρτητης κινητικότητας.

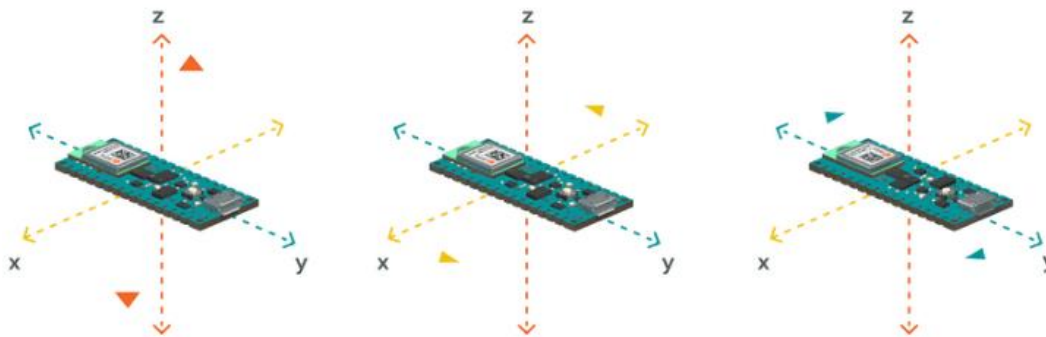
ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

Για την παρούσα εργασία, έχουν χρησιμοποιηθεί ορισμένες τεχνικές μέθοδοι, οι οποίες θεωρείται απαραίτητο να περιγραφούν εκτενώς, προκειμένου να γίνει κατανοητή η ανάλυση που ακολουθεί στις επόμενες σελίδες.

Το Arduino υποστηρίζει ηλεκτρονικές συσκευές και λογισμικό, επιτρέποντας σε ανθρώπους από όλο τον κόσμο να έχουν εύκολη πρόσβαση σε προηγμένες τεχνολογίες που αλληλεπιδρούν με το φυσικό κόσμο. (Arduino.cc) Συγκεκριμένα, σε αυτή τη διπλωματική χρησιμοποιείται το Nano 33 BLE Sense, το οποίο βασίζεται στον μικροελεγκτή nRF52840 και λειτουργεί με το λειτουργικό σύστημα Arm Mbed OS. Το Nano 33 BLE Sense διαθέτει τη δυνατότητα σύνδεσης μέσω Bluetooth Low Energy και είναι εξοπλισμένο με αισθητήρες για ανίχνευση χρώματος, απόστασης, κίνησης, θερμοκρασίας, υγρασίας, ήχου και άλλων. (docs.arduino.cc)

Το Bluetooth Low Energy είναι μία εκδοχή της τεχνολογίας Bluetooth Personal Area Network (PAN). Έχει τη δυνατότητα να ελαχιστοποιήσει την κατανάλωση ενέργειας και το κόστος εξοπλισμού στις συσκευές χαμηλής ισχύος, κάτι που το καθιστά μια ανταγωνιστική λύση ανάμεσα σε μια μεγάλη ποικιλία τεχνικών ασύρματης μετάδοσης που ήδη υπάρχουν στην καθημερινή ζωή για πολλές εφαρμογές. (Liu D., Zhang Y., Zhou H., “A Comprehensive Study of Bluetooth Low Energy”, 2021) Το Bluetooth Low Energy περιορίζει την κατανάλωση ενέργειας μειώνοντας τον χρόνο κατά τον οποίο ο ασύρματος δέκτης Bluetooth είναι ενεργοποιημένος. (docs.arduino.cc) Ο τρόπος με τον οποίο λειτουργεί είναι ο εξής: Όταν η σύνδεση Bluetooth δημιουργείται, η περιφερειακή συσκευή εκπέμπει πληροφορίες για τον εαυτό της προς οποιαδήποτε κοντινή συσκευή. Ταυτόχρονα, η κεντρική συσκευή πραγματοποιεί σάρωση και λαμβάνει πληροφορίες για οποιαδήποτε συσκευή ή συσκευές εκπέμπουν. Μόλις η κεντρική συσκευή λάβει τις εκπομπές από την περιφερειακή συσκευή, γίνεται προσπάθεια να συνδεθεί με την περιφερειακή συσκευή. Μόλις δημιουργηθεί η σύνδεση, η κεντρική συσκευή αλληλεπιδρά με τις πληροφορίες που διαθέτει η περιφερειακή συσκευή.

Ένα επιταχυνσιόμετρο είναι ένας ηλεκτρομηχανικός μηχανισμός που χρησιμοποιείται για τη μέτρηση των δυνάμεων επιτάχυνσης. Τέτοιες δυνάμεις μπορεί να είναι στατικές, όπως η συνεχής δύναμη της βαρύτητας, ή, όπως συμβαίνει με πολλές φορητές συσκευές, δυναμικές για να ανιχνεύει την κίνηση ή τις δονήσεις. Στο έξυπνο γάντι χρησιμοποιείται το επιταχυνσιόμετρο τριών αξόνων του LSM9DS1 Inertial Module ως ένα «επίπεδο» που παρέχει πληροφορίες σχετικά με τη θέση της πλακέτας (βλ. Εικόνα 1). Με αυτή την εφαρμογή μπορεί να ανιχνευθεί η σχετική θέση της πλακέτας καθώς και τις μοίρες, κλίνοντας την πλακέτα προς τα πάνω, κάτω, αριστερά ή δεξιά. (docs.arduino.cc)



Εικόνα 1 Λειτουργία επιταχυνσιόμετρου

Η αναγνώριση χειρονομιών επιτυγχάνεται μέσω του ενσωματωμένου αισθητήρα APDS9960. Ο αισθητήρας APDS9960 είναι μία πολυλειτουργική συσκευή που διαθέτει προηγμένη ανίχνευση χειρονομιών, ανίχνευση εγγύτητας, ψηφιακή ανίχνευση περιβαλλοντικού φωτός (ALS) και ανίχνευση χρώματος (RGBC). Η ανίχνευση χειρονομιών του αισθητήρα χρησιμοποιεί τέσσερις φωτοανιχνευτές προς τις τέσσερις κατευθύνσεις για να ανιχνεύσει την ανακλαστική υπέρυθη ενέργεια (που προέρχεται από το ενσωματωμένο LED) και να μετατρέψει τις φυσικές πληροφορίες κίνησης (δηλαδή ταχύτητα, κατεύθυνση και απόσταση) σε ψηφιακές πληροφορίες. (docs.arduino.cc)

Η βαρομετρική πίεση μετρείται από τον ενσωματωμένο αισθητήρα LPS22HB. Ο LPS22HB είναι ένας υπερ-συμπαγής πιεζοανθεκτικός αισθητήρας απόλυτης πίεσης που λειτουργεί ως βαρόμετρο με ψηφιακή έξοδο. Η συσκευή αποτελείται από ένα στοιχείο αίσθησης και μια διεπαφή IC που επικοινωνεί μέσω I2C ή SPI από το στοιχείο αίσθησης στην εφαρμογή. Το στοιχείο αίσθησης ανιχνεύει την απόλυτη πίεση και αποτελείται από μία ανεσταλμένη μεμβράνη από πυρίτιο και λειτουργεί σε ένα εύρος θερμοκρασίας από -40°C έως $+85^{\circ}\text{C}$. (docs.arduino.cc)

Στο Arduino Nano 33 BLE Sense η μέτρηση και εκτύπωση τιμών θερμοκρασίας και υγρασίας γίνεται δυνατή από τον ενσωματωμένο αισθητήρα HTS221. Οι αισθητήρες θερμοκρασίας είναι εξαρτήματα που μετατρέπουν τη φυσική θερμοκρασία σε ψηφιακά δεδομένα. Αντίστοιχα, οι αισθητήρες υγρασίας μπορούν να μετρήσουν τα επίπεδα ατμοσφαιρικής υγρασίας και να μεταφράσουν αυτά τα δεδομένα σε ηλεκτρικό σήμα. Οι αισθητήρες θερμοκρασίας και υγρασίας είναι απαραίτητοι για την περιβαλλοντική παρακολούθηση, ειδικά μέσα και γύρω από ευαίσθητο ηλεκτρονικό εξοπλισμό. (docs.arduino.cc)

Η τεχνητή νοημοσύνη πρόκειται για μια επιστημονική ανάπτυξη μέσω της οποίας μια μηχανή δημιουργείται για να εκτελεί δράσεις που εμπίπτουν στο πεδίο της ανθρώπινης νοημοσύνης, όπως η μάθηση, η λογική, η αυτοδιόρθωση και η αυτοπρογραμματισμός. Ο κύριος στόχος της τεχνητής νοημοσύνης είναι να καταστήσει τους υπολογιστές ικανούς να εκτελούν το έργο που κάνουν οι άνθρωποι. (Karan Aggarwal et al., Iraqi Journal for Computer Science and Mathematics, 2022)

Η Μηχανική Μάθηση (Machine Learning, ML) είναι ένας κλάδος της Τεχνητής Νοημοσύνης στον οποίο οι μηχανές διδάσκονται να επιλύουν ένα πρόβλημα δημιουργώντας αυτόνομα και αυτόματα προγράμματα. Οι μηχανές «μαθαίνουν» να επεξεργάζονται μεγάλο όγκο

δεδομένων κατά τη διάρκεια αυτού που συνήθως αποκαλείται «φάση εκπαίδευσης». Κατά τη φάση της εκπαίδευσης, η μηχανή «τιμωρείται» για κάθε εσφαλμένο αποτέλεσμα που παράγει ανάλογα με τα συγκεκριμένα δεδομένα εισόδου. Με το πέρασμα του χρόνου, η μηχανή γίνεται καλύτερη στην παραγωγή των σωστών δεδομένων εξόδου. Όλη η Μηχανική Μάθηση ουσιαστικά συνίσταται σε αυτήν τη διαδικασία διόρθωσης του μοντέλου μέχρι να επιτευχθεί το επιθυμητό επίπεδο απόδοσης. (Cherian, 2023)

ΟΡΙΣΜΟΣ ΠΡΟΒΛΗΜΑΤΟΣ

Η παρούσα διπλωματική εργασία ασχολείται με το πρόβλημα της επικοινωνίας και της αυτονομίας των ατόμων με οπτικές και κινητικές δυσκολίες και έχει ως στόχο τη δημιουργία ενός έξυπνου γαντιού με δυνατότητες αναγνώρισης κινήσεων και φωνητικών εντολών. Το πρόβλημα που επιλύεται είναι η περιορισμένη ικανότητα αυτών των ατόμων να επικοινωνήσουν και να αλληλεπιδράσουν με το περιβάλλον τους λόγω των κινητικών τους δυσκολιών. Αυτές οι δυσκολίες έχουν σημαντικές επιπτώσεις στην ποιότητα ζωής τους και την ανεξαρτησία τους.

Πιο αναλυτικά, οι παραδοσιακές μέθοδοι επικοινωνίας και ελέγχου συχνά απαιτούν τη χρήση των χεριών και των δακτύλων, κάτι που δυσκολεύει σημαντικά τα άτομα με κινητικά προβλήματα. Αυτό μπορεί να προκαλέσει αίσθημα απομόνωσης και εξάρτησης από τους γύρω τους, καθώς δεν μπορούν να εκφράσουν τις ανάγκες τους ή να αναλάβουν τον έλεγχο του περιβάλλοντός τους όπως θα ήθελαν.

Για την αντιμετώπιση αυτού του προβλήματος, αναπτύσσεται ένα έξυπνο γάντι που επιτρέπει στα άτομα με οπτικές και κινητικές δυσκολίες να επικοινωνούν, να ελέγχουν ηλεκτρονικές συσκευές και να αναγνωρίζουν κινήσεις με φυσικό τρόπο. Το γάντι είναι εξοπλισμένο με αισθητήρες κίνησης και παρέχει τη δυνατότητα αναγνώρισης των κινήσεων των δακτύλων και του χεριού. Επιπλέον, περιλαμβάνει μικρόφωνο για τη φωνητική επικοινωνία και την αναγνώριση φωνητικών εντολών.

ΣΧΕΔΙΑΣΗ ΚΑΙ ΚΑΤΑΣΚΕΥΗ ΥΛΙΚΟΥ ΜΕΡΟΥΣ (HARDWARE)

Σενάρια Λειτουργίας

Σενάριο 1

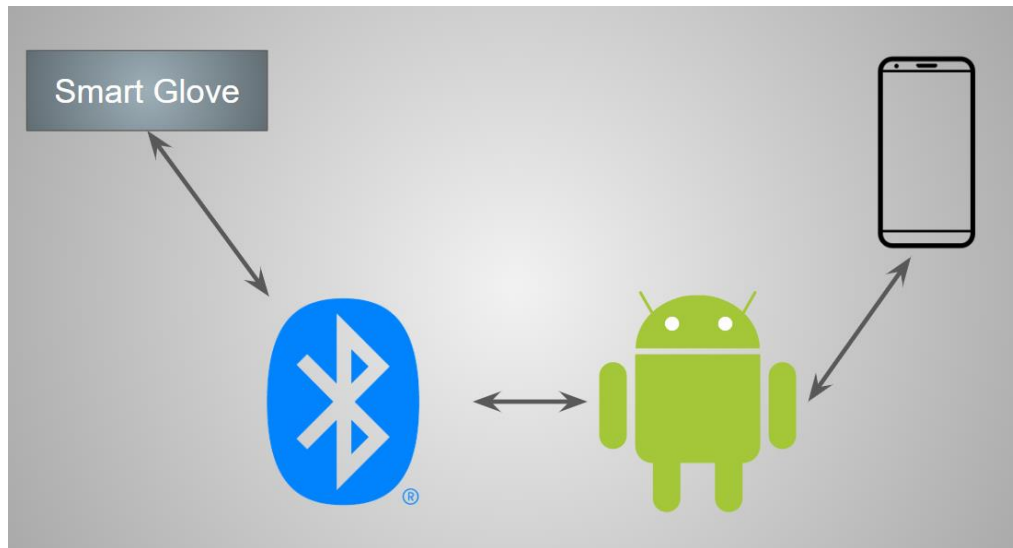
Με την ενεργοποίηση του γαντιού γίνεται και η εκκίνηση των αισθητήρων. Μετά το πέρασμα πέντε δευτερολέπτων, το γάντι αναμένει ηχητική εντολή για να εκκινήσει τη σύνδεση μέσω Bluetooth. Μόλις λάβει την εντολή, τα LED ανάβουν κυκλικά ένα-ένα ανά 0,3 δευτερόλεπτα σε πράσινο χρώμα και μετά αναβοσβήνουν 3 φορές. Έτσι, η εκκίνηση του γαντιού έχει ολοκληρωθεί και είναι έτοιμο να συνδεθεί με την εφαρμογή.

Η σύνδεση του γαντιού με την εφαρμογή γίνεται πατώντας το κουμπί «Connect» στο κάτω μέρος της εφαρμογής. Η εφαρμογή επιλέγει αυτόματα το γάντι χωρίς να χρειάζεται να γίνει επιλογή της συσκευής. Μόλις γίνει η σύνδεση, στο κουτί «Status» της εφαρμογής εμφανίζεται η ένδειξη «Connected» και ταυτόχρονα ενεργοποιείται η χρήση των κουμπιών για τον χειρισμό των LED, τα οποία όσο η συσκευή δεν είναι συνδεδεμένη, είναι απενεργοποιημένα. Παράλληλα στο γάντι, μόλις επιτευχθεί η σύνδεση, τα LED ανάβουν κυκλικά ένα-ένα ανά 0,3 δευτερόλεπτα σε μπλε χρώμα και μετά αναβοσβήνουν 3 φορές.

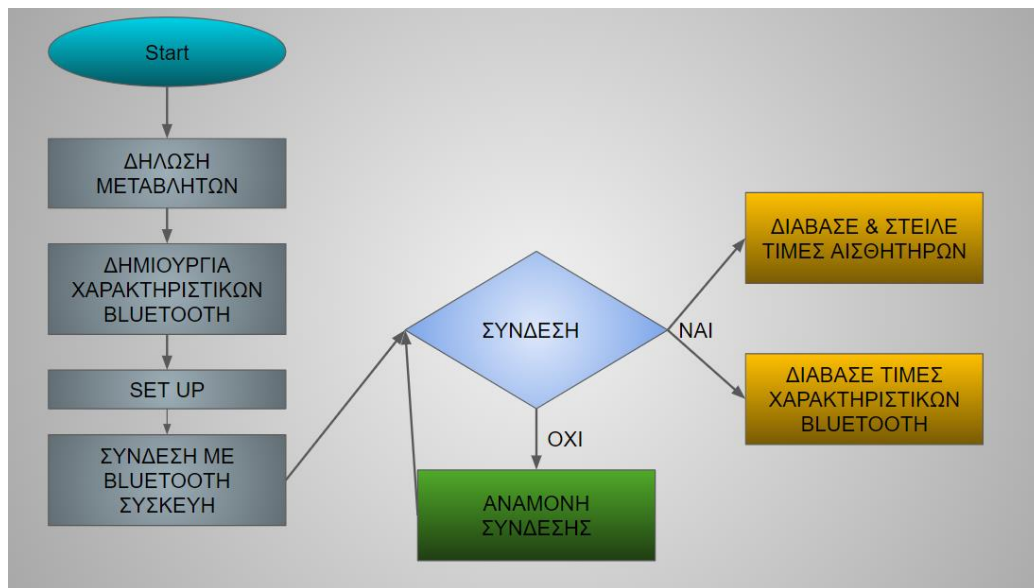
Σε αυτό το σημείο η επικοινωνία γαντιού και εφαρμογής είναι ενεργή και μετά το πέρασμα των 5 πρώτων δευτερολέπτων ξεκινάει η μετάδοση των δεδομένων από το γάντι προς την εφαρμογή. Τα δεδομένα σχετικά με τους αισθητήρες περιβάλλοντος (θερμοκρασία, ατμοσφαιρική πίεση, υγρασία) αποστέλλονται κάθε 10 δευτερόλεπτα, ενώ όλοι οι υπόλοιποι αισθητήρες λειτουργούν και αποστέλλουν δεδομένα σε πραγματικό χρόνο. Οποιαδήποτε αλλαγή γίνεται, μεταδίδεται στην εφαρμογή και ο χρήστης γνωρίζει ακριβώς τη θέση του γαντιού ως προς τις κλίσεις, αν έχει κλειστό κάποιο δάκτυλο ή και όλα, καθώς και το επίπεδο θορύβου στον χώρο. Παράλληλα, ο χρήστης μπορεί να γνωρίζει αν κάποιο αντικείμενο έχει πλησιάσει στο γάντι, αν έχει κάνει κάποια χειρονομία (πάνω, κάτω, δεξιά, αριστερά), αλλά και το χρώμα που βρίσκεται απέναντι από την πάνω μεριά του γαντιού.

Η εφαρμογή, εκτός από τα δεδομένα που δέχεται, μπορεί να ελέγξει το φωτισμό σχετικά με το χρώμα των LED που βρίσκεται στην πλακέτα πάνω στο γάντι. Υπάρχουν 8 επιλογές βασικών χρωμάτων με το πράσινο να είναι προ-ενεργοποιημένο. Εκτός από τα χρώματα, μπορεί να ελέγξει ξεχωριστά αν κάποιο LED θα είναι αναμμένο ή σβηστό, να τα ανάψει όλα μαζί ή ένα-ένα, ακόμη και να τα σβήσει όλα.

Όταν ο χρήστης επιθυμεί να διακόψει τη σύνδεση ή αν η επικοινωνία χαθεί, η εφαρμογή επιστρέφει στην αρχική της κατάσταση και στο κουτί «Status» εμφανίζεται η ένδειξη «No connection».



Εικόνα 2 Επικοινωνία Γάντι - Εφαρμογή



Εικόνα 3 Λειτουργία κώδικα Arduino

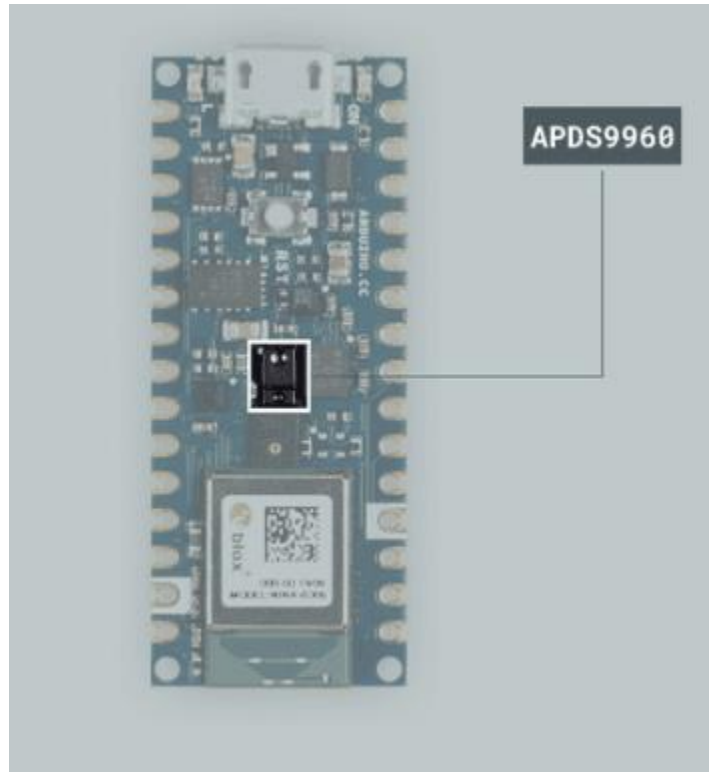
Σενάριο 2

Ο δεύτερος τρόπος χρήσης που έχει το γάντι και μπορεί να συνδυαστεί με τον πρώτο είναι η αναγνώριση φωνητικών εντολών. Ο λόγος που δεν λειτουργούν συνδυαστικά τα δύο σενάρια οφείλεται στην έλλειψη μνήμης που απαιτείται σε τέτοιες εφαρμογές. Σε αυτή τη λειτουργία το γάντι έχει εκπαιδευτεί να αναγνωρίζει τις λέξεις «yes» και «no» και να ανάβει αντίστοιχα πράσινα και κόκκινα LED τα οποία παραμένουν σβηστά σε κάθε άλλη λέξη.

Λειτουργικά Μέρη Κατασκευής

Για την επίτευξη των παραπάνω λειτουργιών έχει χρησιμοποιηθεί ένα Arduino Nano 33 BLE Sense που αποτελεί τον πυρήνα της κατασκευής με μικροελεγκτή nRF52480 και λειτουργικό σύστημα Arm Mbed OS. Ο μικροελεγκτής nRF52480 της Nordic είναι ένα SoC εξαιρετικά χαμηλής κατανάλωσης, ασύρματης επικοινωνίας 2,4 GHz που ενσωματώνει ένα πολύ-πρωτόκολλο πομποδέκτη 2,4 GHz, έναν επεξεργαστή Arm Cortex-M4F και μνήμη flash. (infocenter.nordicsemi.com). Για τη λήψη των περιβαλλοντικών μετρήσεων χρησιμοποιούνται οι αισθητήρες LPS22HB και HTS221. Ο αισθητήρας LPS22HB (βλ. Εικόνα 5) λειτουργεί ως βαρόμετρο και ο αισθητήρας HTS221 (βλ. Εικόνα 6) μετρά και εμφανίζει δεδομένα για την υγρασία και τη θερμοκρασία. (docs.arduino.cc) Για τον HTS221 χρησιμοποιείται το πρωτόκολλο I2C για την επικοινωνία με τον αισθητήρα. Ο αισθητήρας υγρασίας έχει ακρίβεια $\pm 3.5\%$ rH, 20 έως +80% rH και εύρος 0 έως 100%. Ο αισθητήρας θερμοκρασίας έχει ακρίβεια ± 0.5 °C, 15 έως +40 °C και εύρος -40 έως 120°C.

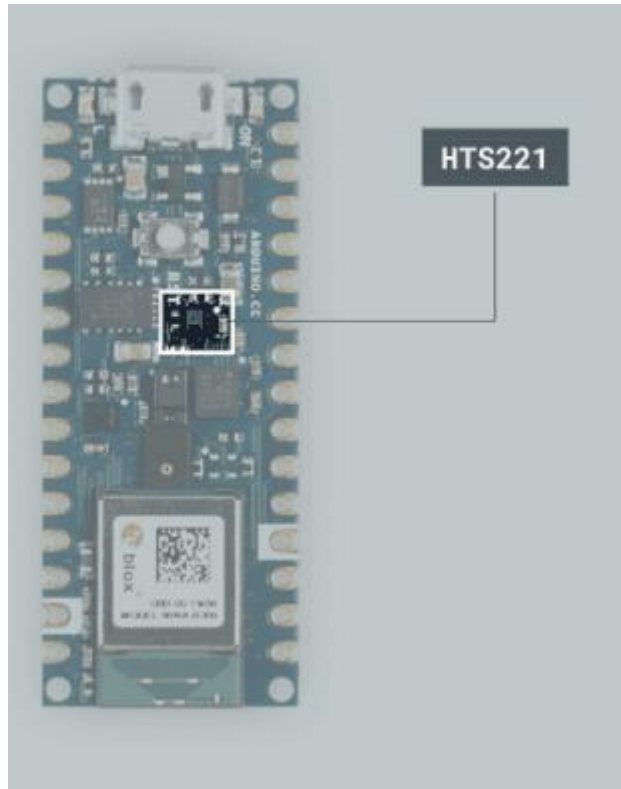
Για την αναγνώριση χρωμάτων, χειρονομίας και απόστασης από τη συσκευή χρησιμοποιείται ο αισθητήρας APDS9960 (βλ. Εικόνα 4). Χρησιμοποιείται ο αισθητήρας APDS9960 για να αναγνωρίζονται οι χειρονομίες και να ελέγχεται αντίστοιχα το RGB LED της πλακέτας. Επιπλέον, η πλακέτα προγραμματίστηκε να αναβοσβήνει το ενσωματωμένο LED και να αλλάζει τα χρώματα του RGB LED ανάλογα με την κατεύθυνση των καθορισμένων χειρονομιών. Για την αναγνώριση φωνητικών εντολών γίνεται η χρήση του αισθητήρα MP34DT05 που είναι ένα πανκατευθυντικό μικρόφωνο για τη λήψη και την ανάλυση ήχου. Η αναγνώριση κινήσεων του καρπού γίνεται με τον αισθητήρα LSM9DS1 (βλ. Εικόνα 7) ο οποίος έχει επιταχυνσιόμετρο, μαγνητόμετρο και γυροσκόπιο. Όλοι οι παραπάνω αισθητήρες βρίσκονται ενσωματωμένοι στον ελεγκτή. Για την αναγνώριση του κλειστού δακτύλου χρησιμοποιείται αγωγήμνη ταινία χαλκού σε κάθε δάκτυλο και στην παλάμη, ο τρόπος λειτουργίας είναι ίδιος με ένα διακόπτη. Οι φωτεινές ενδείξεις LED είναι τεχνολογίας WS2812b, δηλαδή είναι addressable LEDs. Για την αποστολή των δεδομένων γίνεται η χρήση τεχνολογίας Bluetooth 5 Low Energy, τεχνολογία που βρίσκεται ενσωματωμένη στον ελεγκτή. Η διεπαφή με τον χρήστη γίνεται μέσω μιας εφαρμογής Android, η οποία έχει αναπτυχθεί στο App Inventor και προβάλλει γραφικά όλα τα δεδομένα που αποστέλλονται.



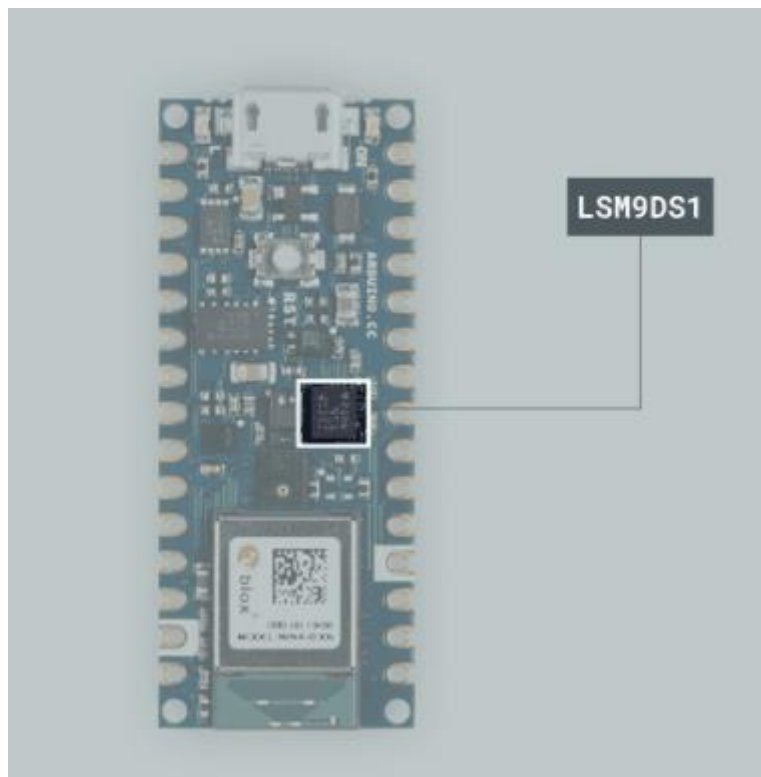
Εικόνα 4 Αισθητήρας APDS9960



Εικόνα 5 Αισθητήρας LPS22HB



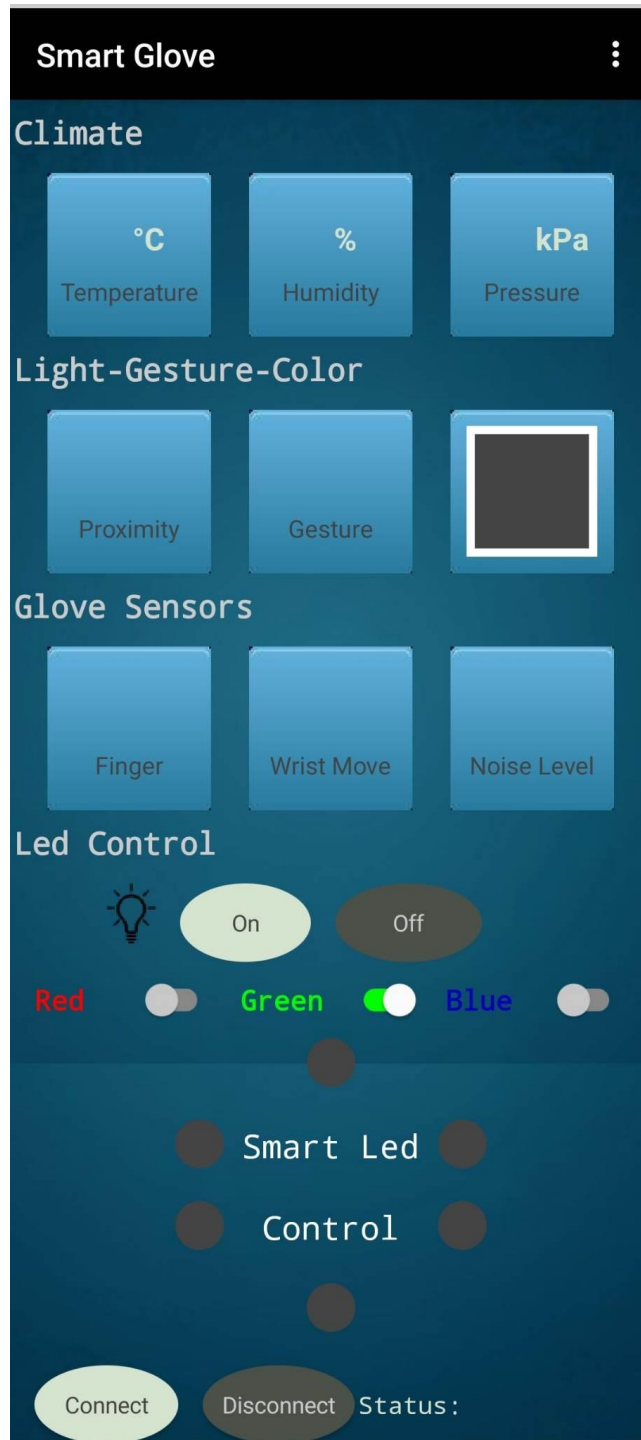
Εικόνα 6 Αισθητήρας HTS221



Εικόνα 7 Αισθητήρας LSM9DS1

Τα καλώδια που ενώνονται με το δείκτη, το μέσο και τον παράμεσο οδηγούνται το καθένα ξεχωριστά σε ένα διαφορετικό αναλογικό ακροδέκτη του ελεγκτή. Ο αντίχειρας και η παλάμη είναι συνδεδεμένα στο «ground». Οποιοδήποτε δάχτυλο ακουμπήσει στον αντίχειρα ή στην παλάμη τότε κλείνει το κύκλωμα και η τιμή που λαμβάνει ο ελεγκτής από τον αντίστοιχο ακροδέκτη είναι τα 0. Έτσι καταλαβαίνουμε στην πράξη ποιο δάχτυλο ή ποιος συνδυασμός δακτύλων έχει κλείσει.





Εικόνα 8 Διεπαφή χρήστη με διαδικτυακή εφαρμογή

Smart Glove

Climate

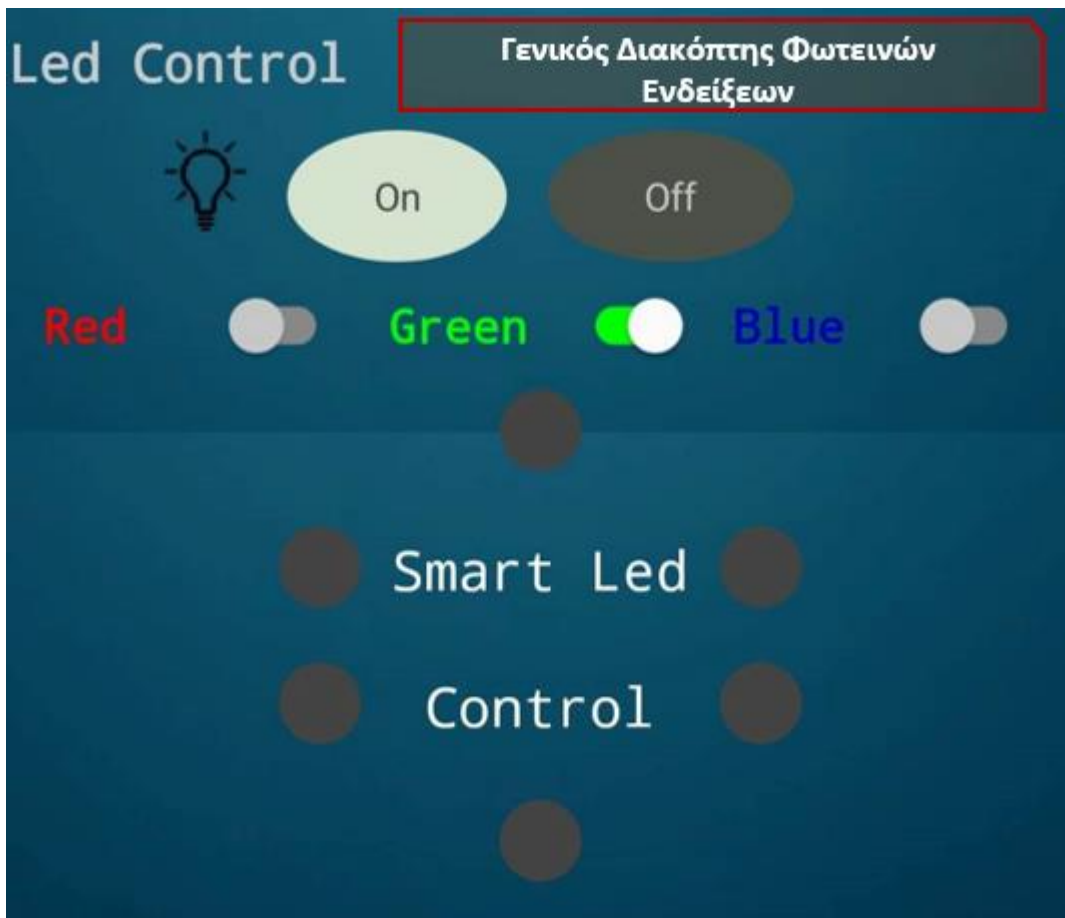
Θερμοκρασία °C Temperature	Υγρασία % Humidity	Πίεση kPa Pressure
----------------------------------	--------------------------	--------------------------

Light-Gesture-Color

Εγγύτητα Proximity	Κίνηση Gesture	Χρώμα
-----------------------	-------------------	-------

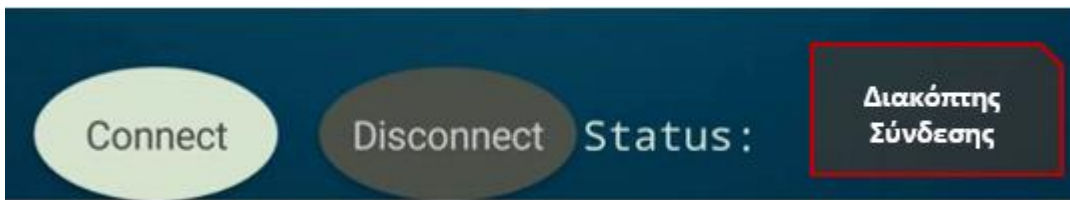
Glove Sensors

Δάκτυλο Finger	Κίνηση Καρπού Wrist Move	Επίπεδο Ήχου Noise Level
-------------------	--------------------------------	--------------------------------



Γενικός Διακόπτης Φωτεινών Ενδείξεων: Για το χειρισμό των φωτεινών ενδείξεων LED υπάρχει ένας γενικός διακόπτης δύο καταστάσεων On/Off όπου με το On ανάβουν όλα τα φώτα και με το Off σβήνουν. Μπορεί να γίνει συνδυασμός χρωμάτων έχοντας σαν κύρια χρώματα το κόκκινο, το πράσινο και το μπλε.

Κάτω από την επιλογή χρώματος υπάρχουν σχηματισμένα τα LED όπως είναι και στην πλακέτα όπου ανάβει το κάθε ένα ξεχωριστά ανάλογα με την κίνηση που πραγματοποιείται. Όταν το χέρι είναι τοποθετημένο επίπεδα είναι όλα τα LED σβηστά. Ανάλογα με τις κινήσεις του χεριού ανάβουν και τα LED αντίστοιχα. Για παράδειγμα, εάν ο χρήστης κάνει μόνο αριστερή κίνηση του χεριού θα ανάψουν τα δύο LED από αριστερά. Εάν, όμως, κάνει αριστερή στροφή και προς τα μπροστά το χέρι, θα ανάψει το επάνω αριστερά LED μόνο. Αυτό γίνεται για να δώσουμε περισσότερη ακρίβεια στις κινήσεις του χεριού. Με αυτό τον τρόπο, συμπεριλαμβάνονται οι βασικές κινήσεις πάνω, κάτω, δεξιά, αριστερά καθώς και οι ενδιάμεσες αυτών συνδυαστικά.

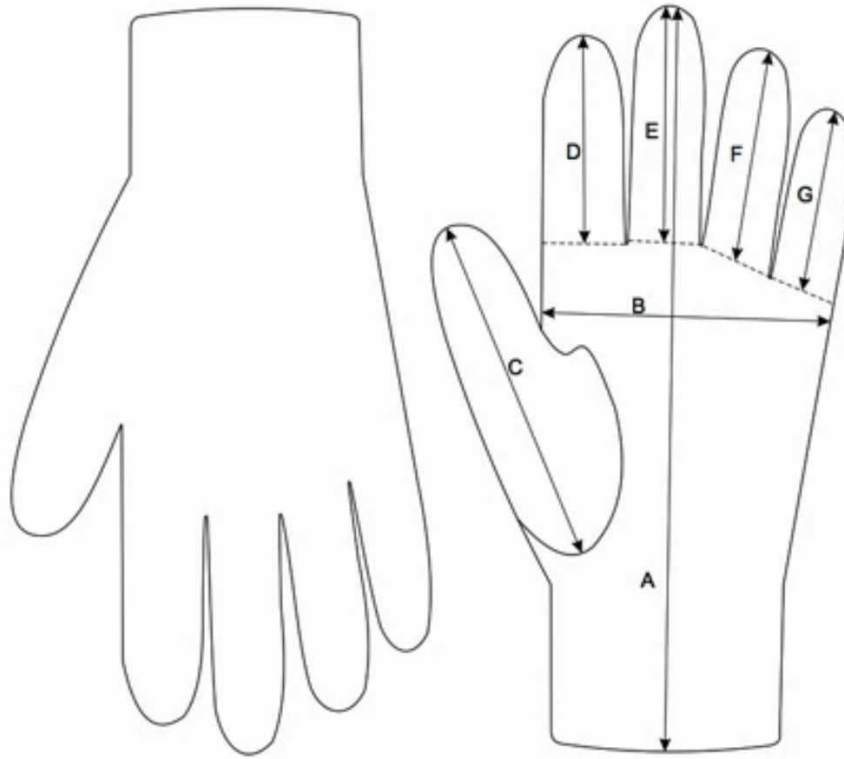


Εικόνα 9 Εικόνες Εφαρμογής με περιγραφή

Το **κουμπί Σύνδεσης/Αποσύνδεσης** αποτελείται από δύο κουμπιά: το διακόπτη σύνδεσης «Connect» και το διακόπτη αποσύνδεσης «Disconnect». Όταν πατηθεί το «Connect», η συσκευή συνδέεται μέσω Bluetooth με το έξυπνο γάντι. Όταν γίνει επιτυχής σύνδεση, υπάρχει αναμμένο και το αντίστοιχο λαμπάκι στην πλακέτα μας. Για να αποσυνδεθεί το έξυπνο γάντι, ο χρήστης πατάει το κουμπί «Disconnect».

Μηχανολογικό σχέδιο

Στην παρακάτω εικόνα (βλ. Εικόνα 10) βλέπουμε την κάτοψη του Μηχανολογικό σχεδίου του γαντιού. Ο πίνακας (βλ. Πίνακας 1) αναγράφει τις διαστάσεις του γαντιού.



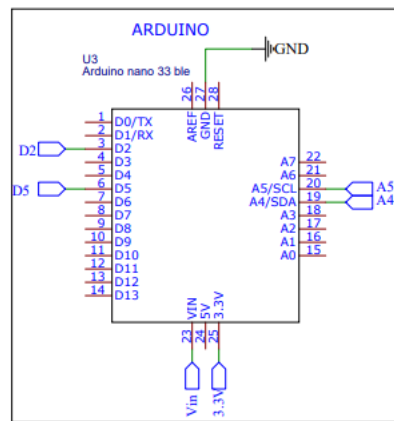
Εικόνα 10 Μηχανολογικό σχέδιο

	Διαστάσεις (cm)
A	22
B	8,5
C	6
D	7,5
E	8
F	7,5
G	6

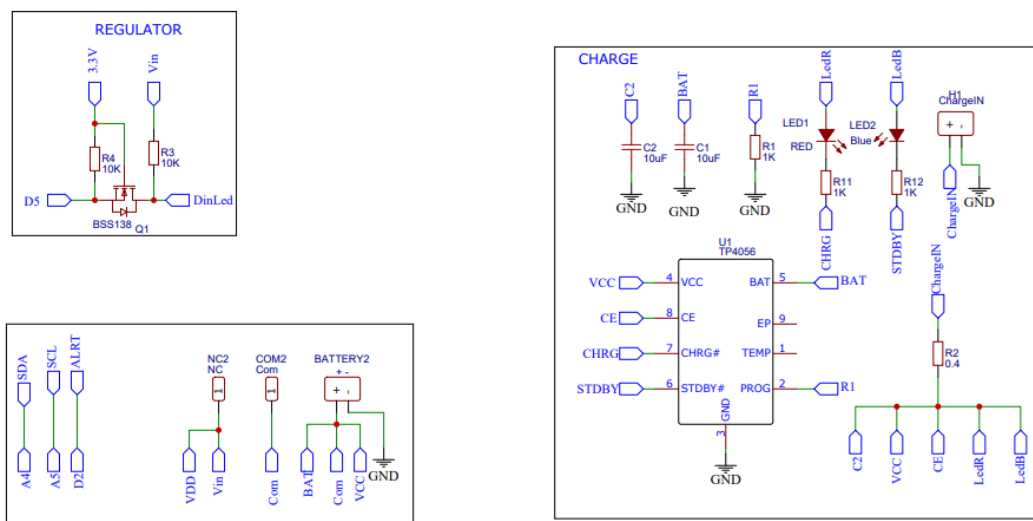
Πίνακας 1 Μηχανολογικό σχέδιο - Πίνακας διαστάσεων

Ηλεκτρονικό κύκλωμα

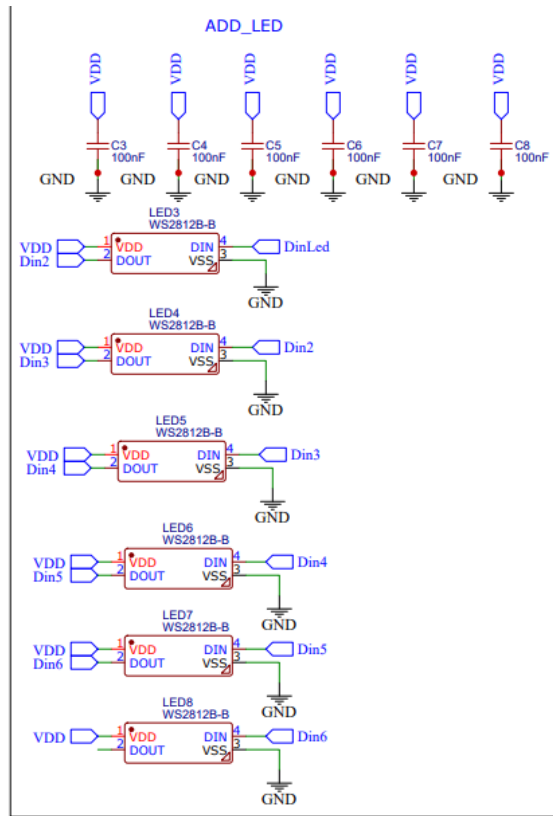
Στην παρακάτω εικόνα (βλ. Εικόνα 11) παρουσιάζεται το σχηματικό διάγραμμα του ηλεκτρονικού σχεδίου το οποίο δημιουργήθηκε μέσω του προγράμματος EasyEDA. Το τυπωμένο κύκλωμα αποτελείται από τον μικροελεγκτή Arduino Nano 33 BLE Sense το οποίο προγραμματίζεται μέσω του open source προγράμματος Arduino IDE. Η γλώσσα που προγραμματίζεται είναι μία παραλλαγή της C++ και μπορεί να προγραμματιστεί και μέσω MicroPython. Η MicroPython είναι μια εφαρμογή της Python και περιλαμβάνει μερικές από τις βασικές βιβλιοθήκες της Python. (docs.arduino.cc) Έχει σχεδιαστεί για μικροελεγκτές με περιορισμένο περιβάλλον. Πάνω στο κύκλωμα υπάρχουν 6 πυκνωτές 10uF για κάθε ένα από τα LED για την μείωση του θορύβου, ένα τρανζίστορ για την εναλλαγή λογικής κατάστασης εξόδου του pin του ελεγκτή από 3.3V σε 5V για την οδήγηση των LED, καθώς και ένα κύκλωμα φόρτισης μπαταρίας TP4056 για μπαταρίες λιθίου. Στα Pin A0-A3 συνδέονται τα δάκτυλα και το κοινό τους καλώδιο συνδέεται απευθείας στο GND.



Εικόνα 11 Έξοδοι Arduino



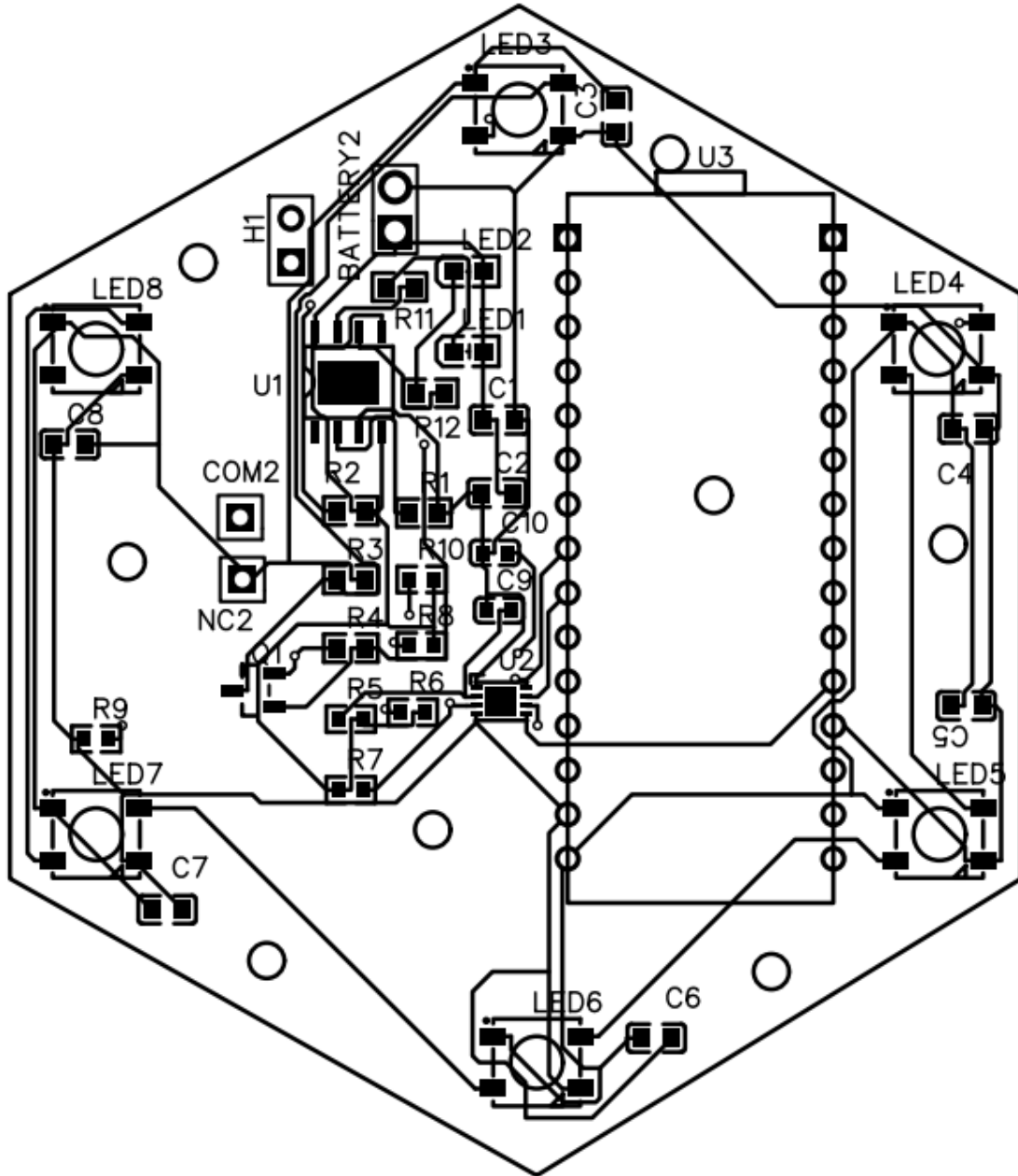
Εικόνα 12 Κυκλώματα φόρτισης, αλλαγή λογικής κατάστασης και σημεία σύνδεσης



Εικόνα 13 Κύκλωμα LED

Τυπωμένο κύκλωμα

Στην παρακάτω εικόνα (βλ. Εικόνα 14) παρουσιάζονται οι λεπτομέρειες του τυπωμένου κυκλώματος.

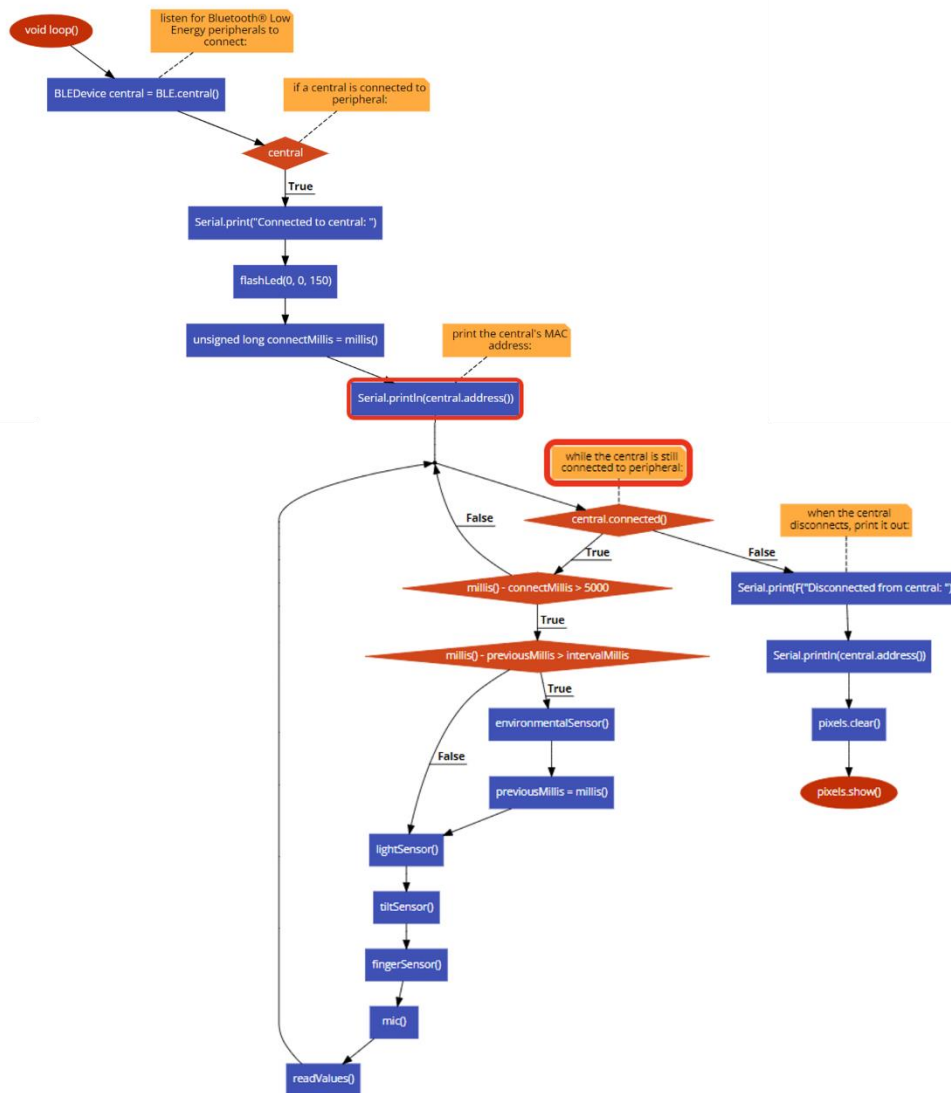


Εικόνα 14 Τυπωμένο Κύκλωμα

ΑΝΑΛΥΣΗ ΚΑΙ ΣΥΓΓΡΑΦΗ ΛΟΓΙΣΜΙΚΟΥ (SOFTWARE)

Διαγράμματα Ροής

- Βρόχος: Στο Βρόχο ελέγχεται η σύνδεση του γαντιού με συσκευή Bluetooth και διαβάζονται οι αισθητήρες.



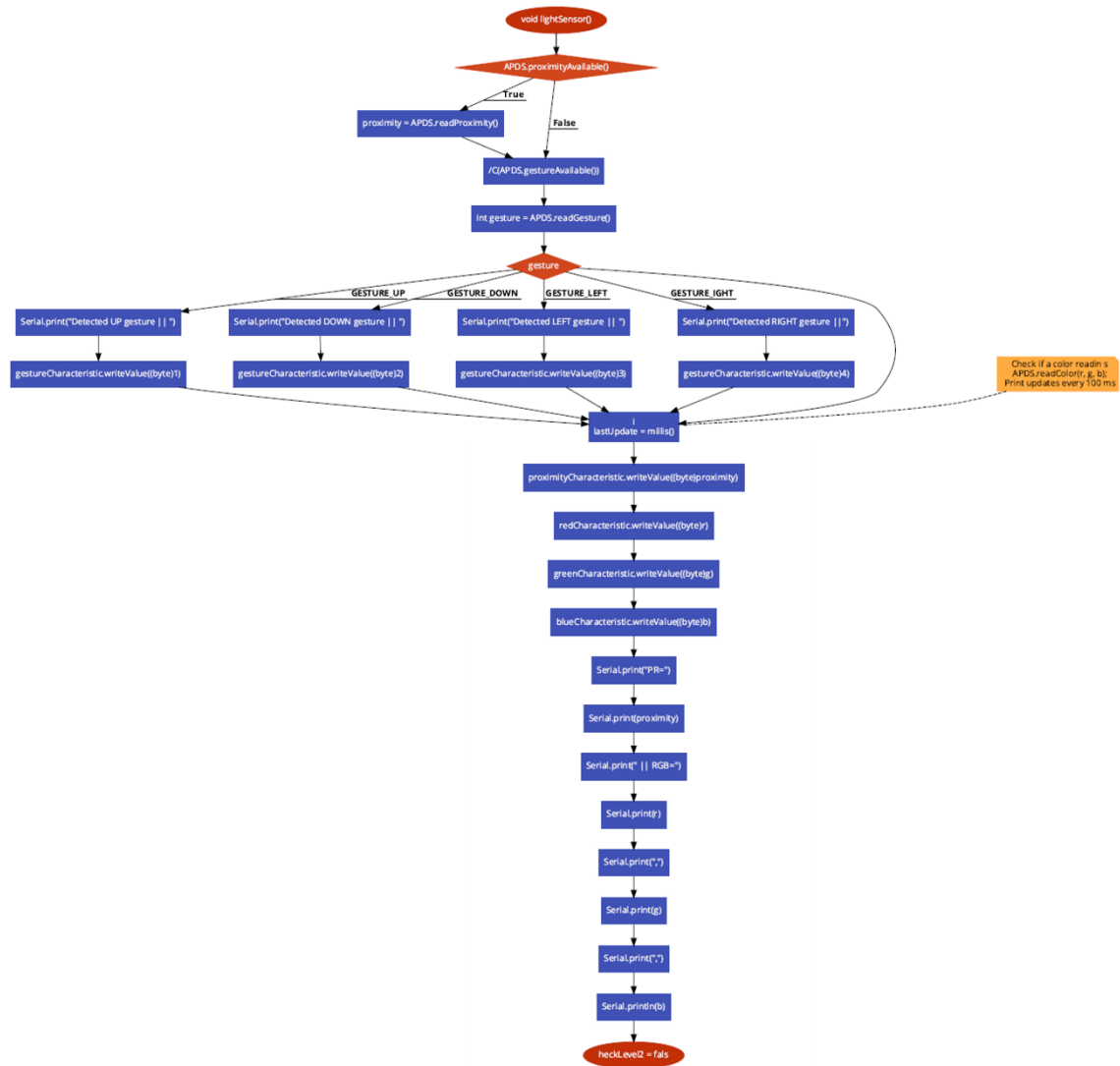
Εικόνα 15 Διάγραμμα Ροής Loop

- Συνάρτηση Περιβαλλοντικών Αισθητήρων: Ανάγνωση Περιβαλλοντικών δεδομένων



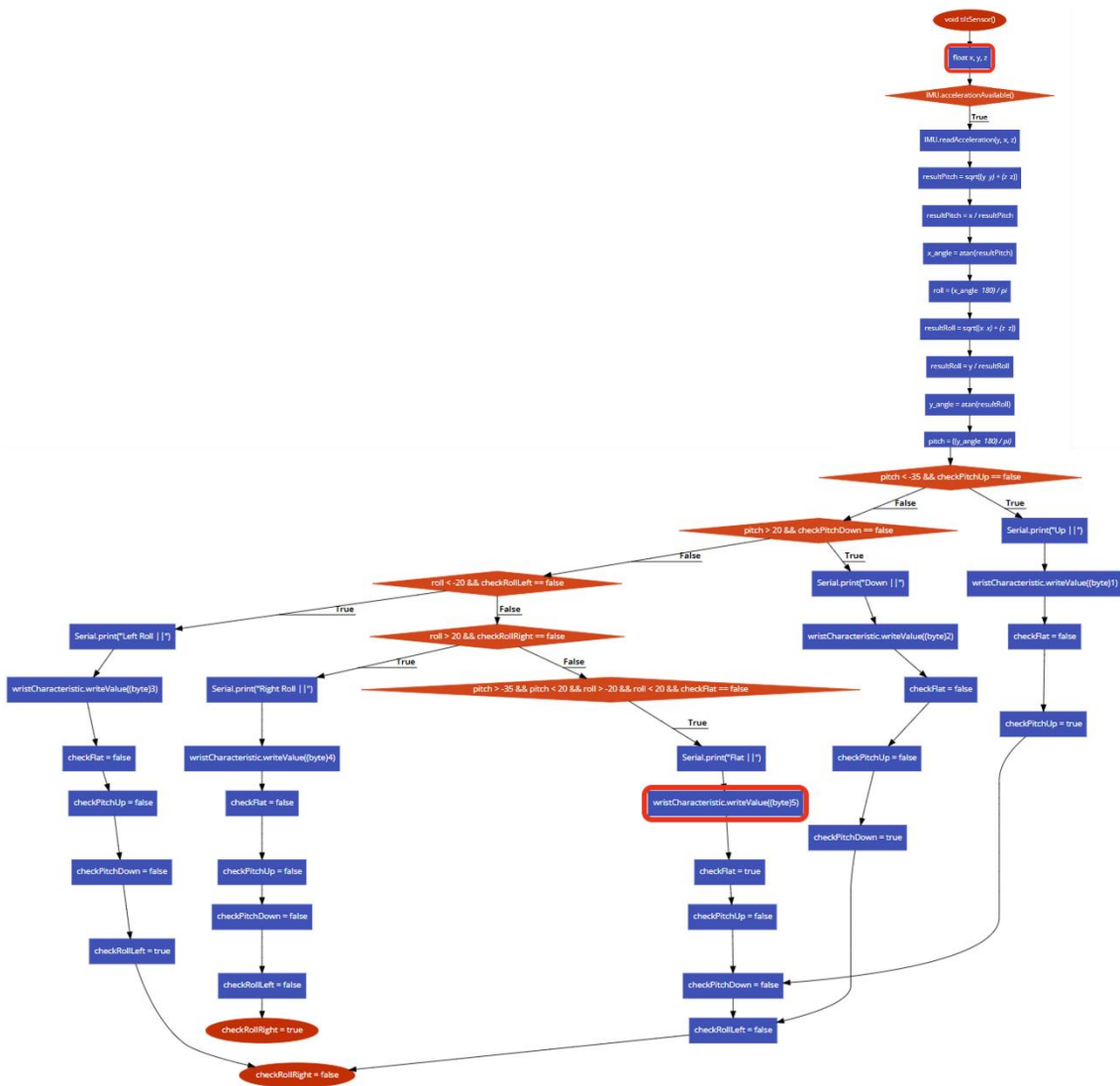
Εικόνα 16 Διάγραμμα Ροής Συνάρτησης Περιβαλλοντικών Αισθητήρων

- Συνάρτηση Οπτικών Αισθητήρων: Ανάγνωση εγγύτητας, κίνησης και χρωμάτων



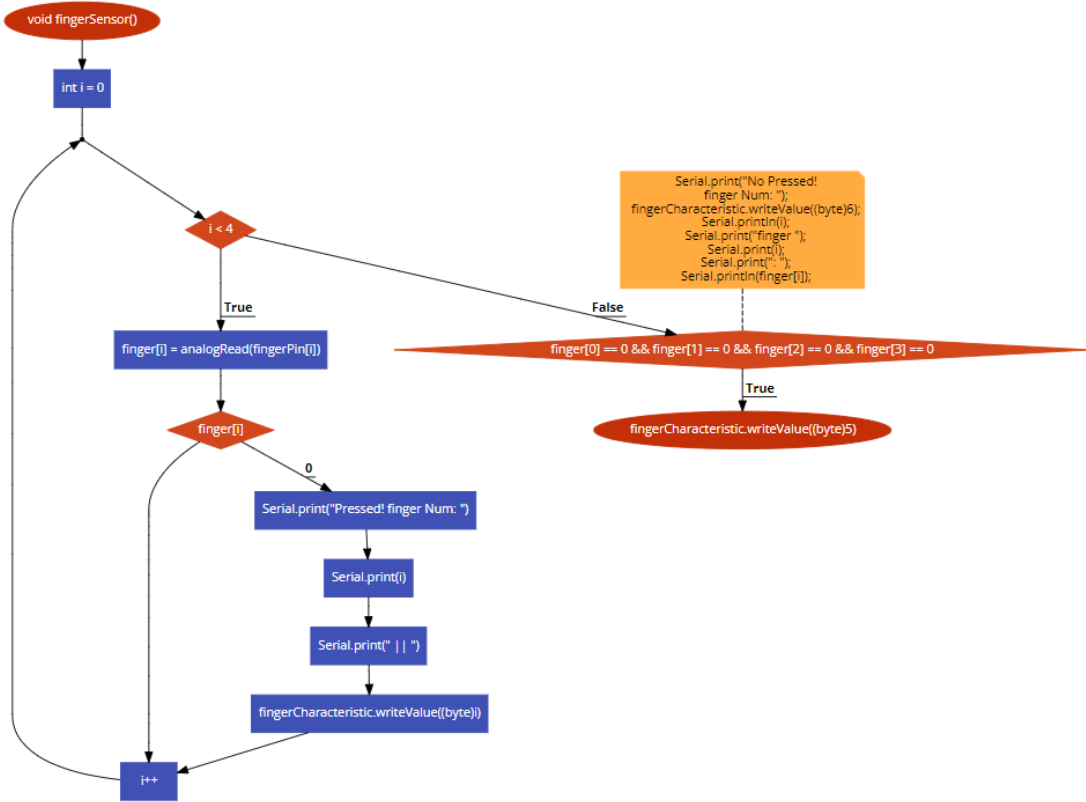
Εικόνα 17 Διάγραμμα Ροής Συνάρτησης Οπτικών Αισθητήρων

- Συνάρτηση Αισθητήρα Κλίσης: Διαβάζει τις τιμές επιτάχυνσης



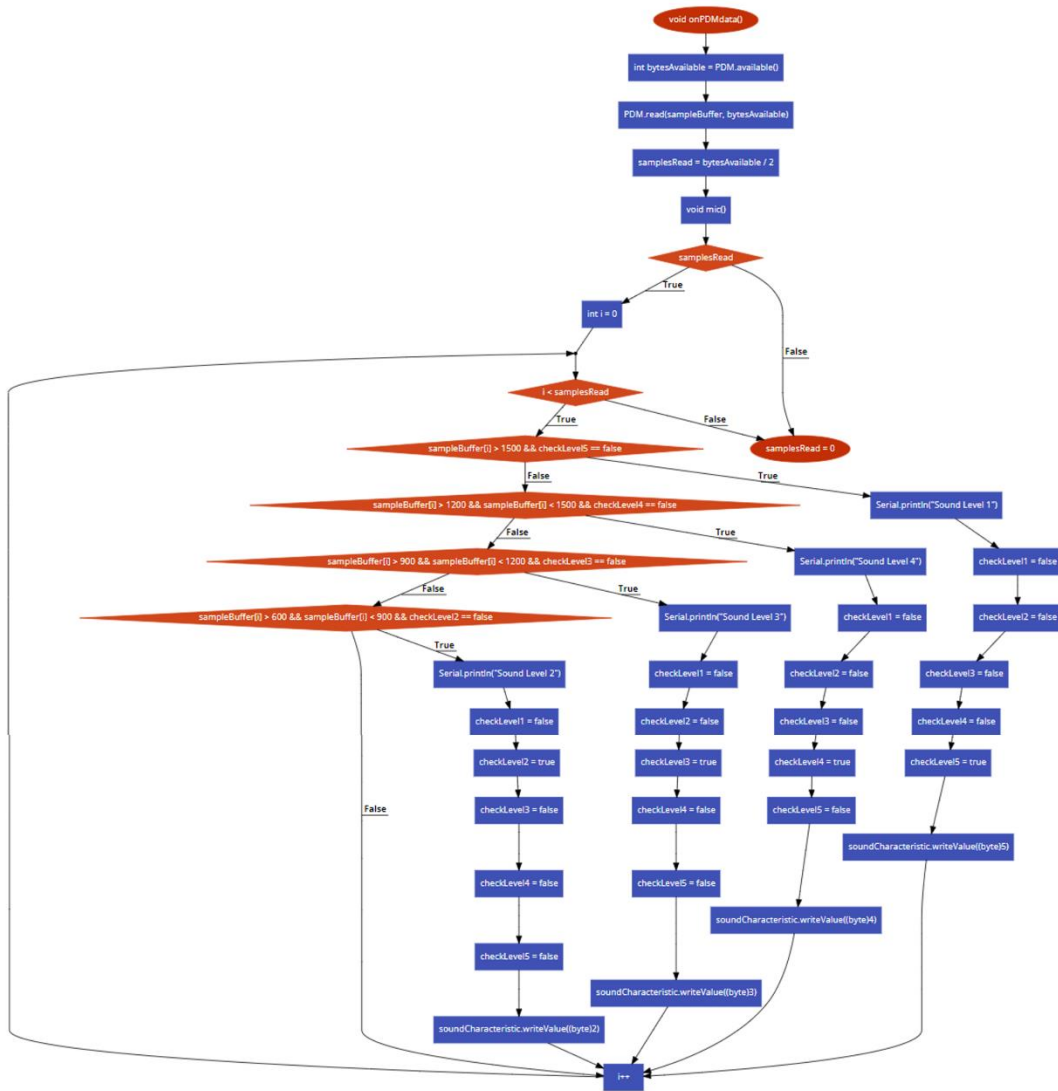
Εικόνα 18 Διάγραμμα Ροής Συνάρτησης Αισθητήρα Κλίσης

- Συνάρτηση Αισθητήρων Δακτύλων: Διαβάζει την πίεση στα δάκτυλα



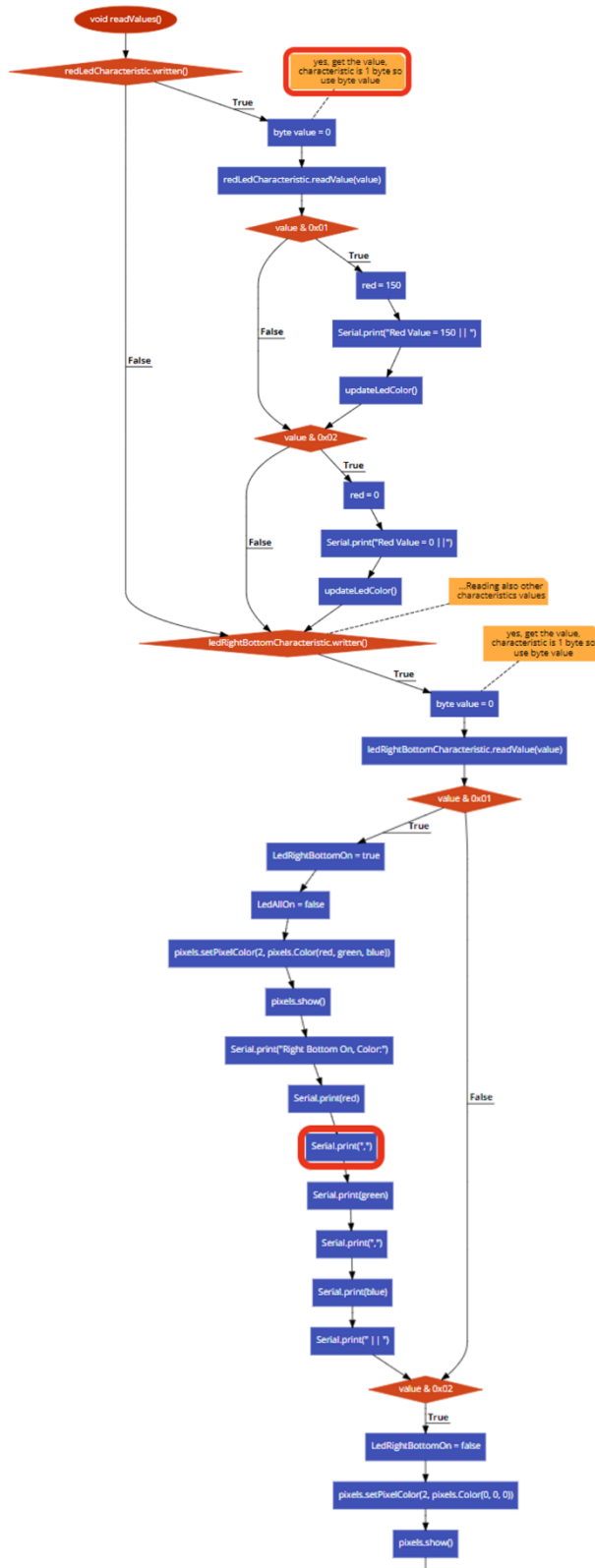
Εικόνα 19 Διάγραμμα Ροής Συνάρτησης Αισθητήρων Δακτύλων

- Συνάρτηση Αισθητήρων Ήχου: Διαβάζει τα δεδομένα ήχου από τον πυκνωτή



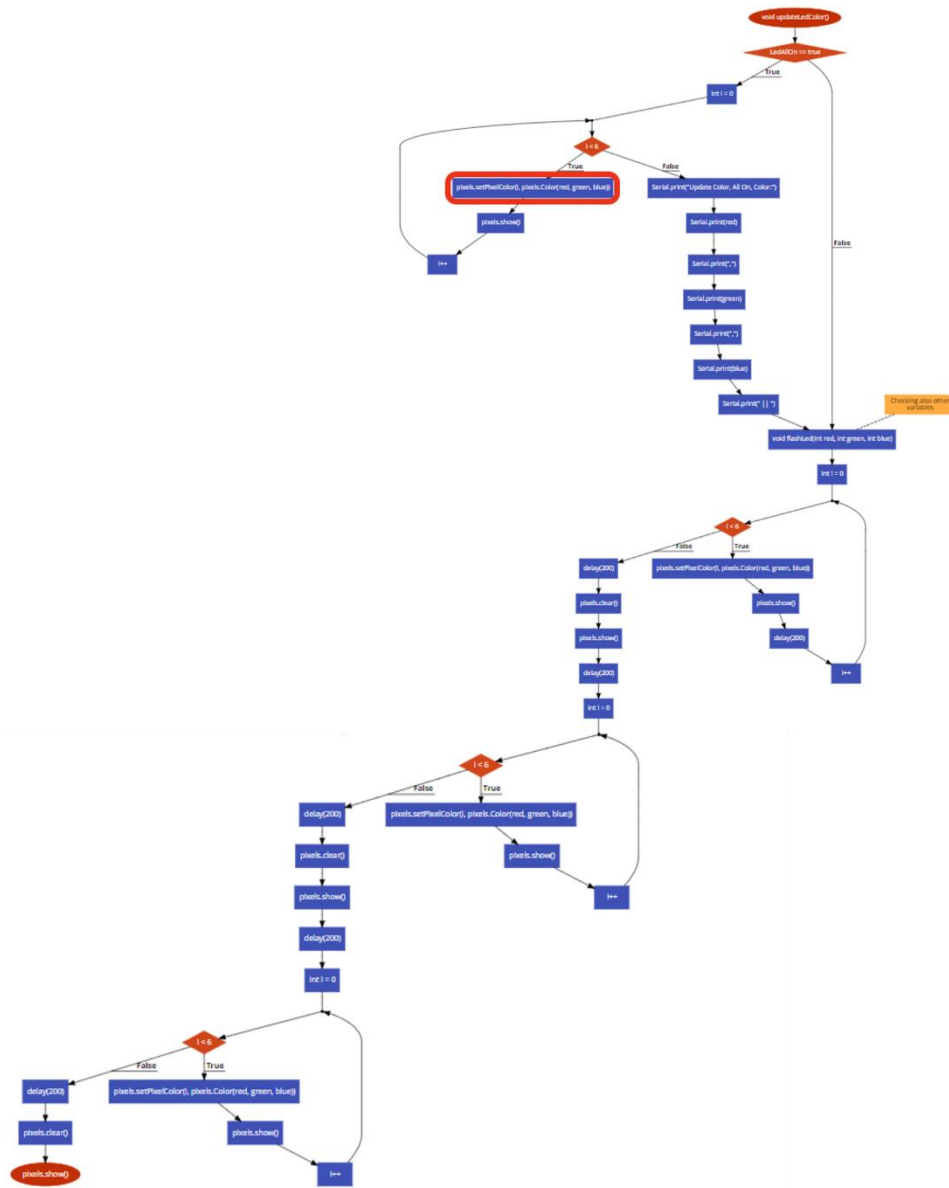
Εικόνα 20 Διάγραμμα Ροής Συνάρτησης Αισθητήρων Ήχου

- Συνάρτηση Διαβάσματος τιμών εισόδου από εφαρμογή: Αναγνωρίζει τις τιμές διάφορων χαρακτηριστικών που προέρχονται από την εφαρμογή



Εικόνα 21 Διάγραμμα Ροής Συνάρτησης Διαβάσματος τιμών εισόδου από εφαρμογή

- Συνάρτηση λειτουργίας LED: ενημερώνει το χρώμα των φωτεινών δεικτών

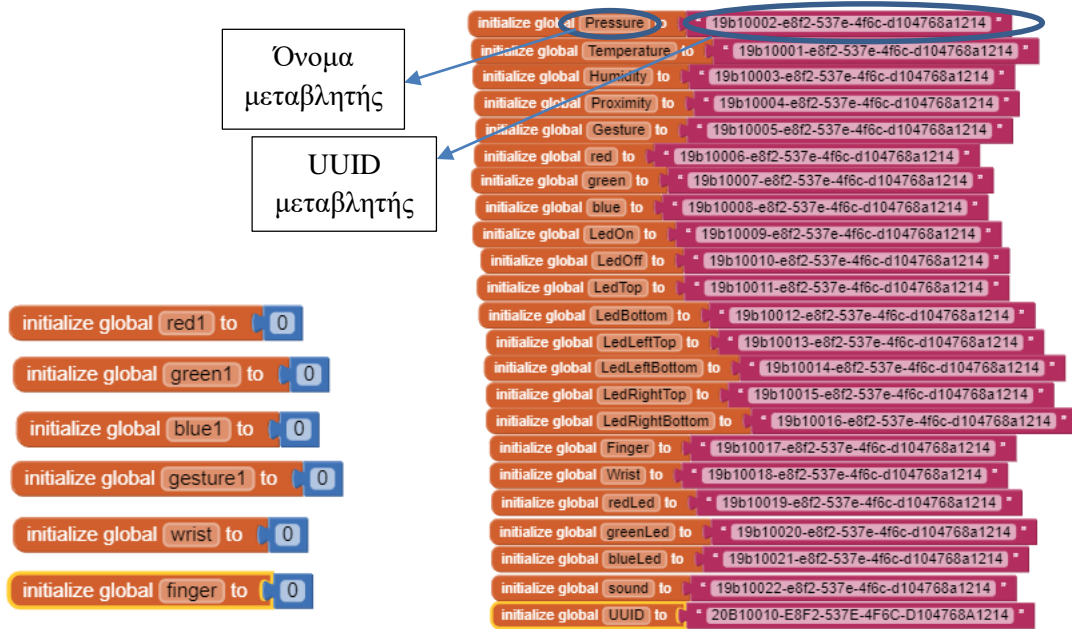


Εικόνα 22 Διάγραμμα Ροής Συνάρτησης λειτουργίας LED

Βασικός Κώδικας Εφαρμογής Android

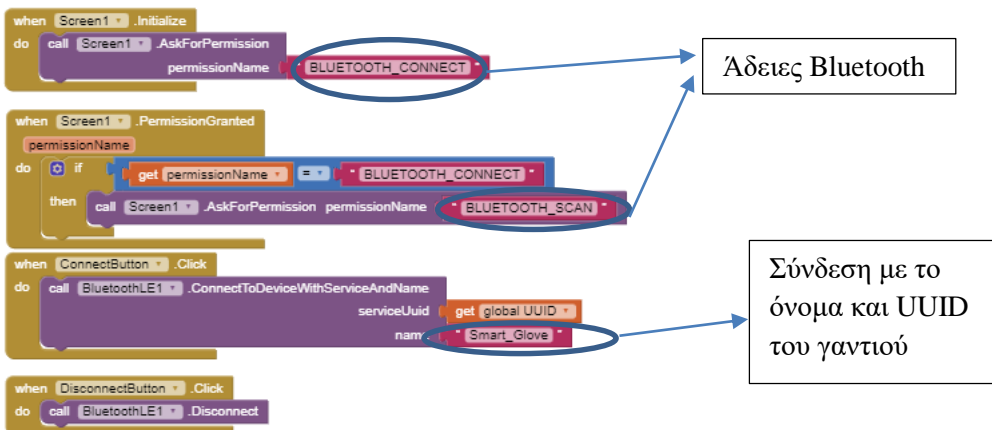
Η εφαρμογή αναπτύχθηκε στο App Inventor. Οι παρακάτω εικόνες δείχνουν τα Block στο backend της εφαρμογής.

-Αρχικοποίηση μεταβλητών για τα ξεχωριστά χαρακτηριστικά Bluetooth και των υπολοίπων μεταβλητών.



Εικόνα 23 Αρχικοποίηση μεταβλητών (Αριστερά) Χαρακτηριστικά Bluetooth (Δεξιά)

-Λειτουργία για κουμπί σύνδεσης, αποσύνδεσης και λειτουργία αποδοχής από τον χρήστη της Android συσκευής αδειών για την χρήση και την ενεργοποίηση των Bluetooth στη συσκευή του.



Εικόνα 24 Άδειες Bluetooth, κουμπί σύνδεσης και κουμπί αποσύνδεσης

-Διεργασίες που γίνονται μόλις επιτευχθεί η σύνδεση μεταξύ γαντιού και συσκευής Android. Ενεργοποίηση κουμπιών για τη χρήση των LED, αλλαγή κειμένου κατάστασης σύνδεσης και εγγραφή στα χαρακτηριστικά για να είναι έτοιμα για επικοινωνία.

```

when BluetoothLE1 .Connected
do
set ConnectButton .Enabled to false
set LedBottom .Enabled to true
set LedLeftBottom .Enabled to true
set LedLeftTop .Enabled to true
set LedOff .Enabled to true
set LedOn .Enabled to true
set LedRightBottom .Enabled to true
set LedRightTop .Enabled to true
set LedTop .Enabled to true
set LedTop .Enabled to true
set Switch1 .Enabled to true
set Switch2 .Enabled to true
set Switch3 .Enabled to true
set StatusValue .Text to "Connected"

```

Εικόνα 25 Ενεργοποίηση κουμπιών μετά τη σύνδεση

```

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Gesture
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global red
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global green
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global blue
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Finger
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Wrist
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Wrist
signed true

call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global sound
signed true

when BluetoothLE1 .Connected
do
set StatusValue .Text to "Connected"
if call BluetoothLE1 .CanReadCharacteristic
serviceUuid get global UUID
characteristicUuid get global Temperature
then
if call BluetoothLE1 .CanRegisterForCharacteristic
serviceUuid get global UUID
characteristicUuid get global Temperature
then
call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Temperature
signed true
call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Pressure
signed true
call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Humidity
signed true
call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Proximity
signed true
call BluetoothLE1 .RegisterForBytes
serviceUuid get global UUID
characteristicUuid get global Gesture
signed true

```

Εικόνα 26 Εγγραφή στα Χαρακτηριστικά Bluetooth

-Διεργασίες που γίνονται μόλις διακοπεί η σύνδεση μεταξύ γαντιού και συσκευής Android. Απενεργοποίηση των κουμπιών και αρχικοποίηση τιμών της εφαρμογής.

```

when BluetoothLE1 - Disconnected
do
  set ConnectButton - Enabled - to true
  set StatusValue - Text - to No connection
  set LedBottom - Enabled - to false
  set LedLeftBottom - Enabled - to false
  set LedLeftTop - Enabled - to false
  set LedOn - Enabled - to false
  set LedOff - Enabled - to false
  set LedRightBottom - Enabled - to false
  set LedRightTop - Enabled - to false
  set LedTop - Enabled - to false
  set LedTop - Enabled - to false
  set Switch1 - Enabled - to false
  set Switch2 - Enabled - to false
  set Switch3 - Enabled - to false
  set FingerValue - Text - to
  set GestureValue - Text - to
  set HumidityValue - Text - to
  set PressureValue - Text - to
  set ProximityValue - Text - to
  set TemperatureValue - Text - to
  set WristValue - Text - to
  set WristValue - Text - to
  set LedBottom - BackgroundColor - to
  set LedLeftBottom - BackgroundColor - to
  set LedLeftTop - BackgroundColor - to
  set LedRightBottom - BackgroundColor - to
  set LedRightTop - BackgroundColor - to
  set LedTop - BackgroundColor - to
  global LedOnPressed - to false
  global LedTopPressed - to false
  global LedTopPressed - to false
  global LedBottomPressed - to false
  global LedBottomPressed - to false
  global LedLeftTopPressed - to false
  global LedLeftTopPressed - to false
  global LedLeftBottomPressed - to false
  global LedLeftBottomPressed - to false
  global LedLeftBottomOnPressed - to false
  global LedRightTopPressed - to false
  global LedRightTopOnPressed - to false
  global LedRightBottomPressed - to false
  global LedRightBottomOnPressed - to false
  global LedBottomPressed - to false
  global LedBottomOnPressed - to false
  set ColorValue - BackgroundColor - to

```

Εικόνα 27 Ενέργειες κατά τη διακοπή σύνδεσης

-Διεργασίες που γίνονται μόλις ληφθούν bytes για την σωστή εμφάνιση της τιμής που λήφθηκε στο σημείο που πρέπει να εμφανιστεί στην εφαρμογή

```
when BluetoothLE1 - BytesReceived
  serviceUuid  characteristicUuid  byteValues
do
  if compare texts get characteristicUuid == get global Temperature
  then set TemperatureValue . Text to select list item list get byteValues
      index 1

  if compare texts get characteristicUuid == get global Pressure
  then set PressureValue . Text to select list item list get byteValues
      index 1

  if compare texts get characteristicUuid == get global Humidity
  then set HumidityValue . Text to select list item list get byteValues
      index 1

  if compare texts get characteristicUuid == get global Proximity
  then set ProximityValue . Text to select list item list get byteValues
      index 1

  if compare texts get characteristicUuid == get global sound
  then set NoiseValue . Text to select list item list get byteValues
      index 1

  if compare texts get characteristicUuid == get global Finger
  then set global finger to select list item list get byteValues
      index 1
  if get global finger == 1
  else if get global finger == 2
  then set FingerValue . Text to "Middle"
  else if get global finger == 3
  then set FingerValue . Text to "Ring"
  else if get global finger == 4
  then set FingerValue . Text to "Little"
  else if get global finger == 5
  then set FingerValue . Text to "Fist"
  else if get global finger == 6
  then set FingerValue . Text to "Open"
  else set FingerValue . Text to ""
```

If a value is true, then do some statements.

The image displays three separate blocks of Scratch code, each implementing a different detection logic. All three blocks start with an 'if' block that compares the 'characteristicUuid' of a selected list item (from a 'byteValues' list) against a specific global variable. If the comparison is successful, the code then sets a global variable to the selected item's index (1) and enters another 'if' block to check the value of that global variable. Depending on the value, the code sets a corresponding 'Text' property of a variable (WristValue, GestureValue, or ColorValue) to a specific string or color name. The first block handles wrist movements (Up, Down, Left, Right, Flat). The second block handles gestures (Up, Down, Left, Right, No Move). The third block handles color detection (red, green, blue) and then uses a 'make a list' block to create a list of the detected colors for the 'ColorValue' variable's background color.

Εικόνα 29 Λειτουργίες όταν ληφθούν bytes από το γάντι

-Λειτουργία κουμπιών για την λειτουργία των LED. Οι παρακάτω εικόνες εξηγούν τα κουμπιά που ενεργοποιούν και απενεργοποιούν όλα τα LED μαζί, οι διακόπτες για την επιλογή χρωμάτων και θα δοθούν δύο παραδείγματα για την λειτουργία των LED του εξάγωνου, καθώς η διαδικασία επαναλαμβάνεται και για τα υπόλοιπα.

The image shows two Scratch code blocks for handling button presses. The left block is for the 'Up' button (LidTop) and the right block is for the 'Down' button (LidBottom). Both blocks start with an 'initialize global' block setting 'LedOnPressed' to false. The main logic is enclosed in a 'when Clicked' event block. Inside, there is an 'if' block that checks if 'global LedOnPressed' is false. If true, it sets 'LedTop' or 'LedBottom' background color to green and 'global LedOnPressed' to true. It then calls 'BluetoothLE1 - WriteBytesWithResponse' with service UUID, characteristic UUID, signed true, and decimal value 1. If the global variable is already true, it sets the background color to grey and 'global LedOnPressed' to false, then calls the same Bluetooth function with signed false and decimal value 2.

Εικόνα 30 Διαδικασία για κομπιά εξάγωνο Πάνο-Κάτω

The image shows three Scratch code blocks for color control, labeled 'redSwitch', 'greenSwitch', and 'blueSwitch'. Each block starts with an 'initialize global' block setting the respective switch to false. The main logic is in a 'when Changed' event block. An 'if' block checks if the global switch is false. If true, it sets the global switch to true and calls 'BluetoothLE1 - WriteBytesWithResponse' with service UUID, characteristic UUID, signed true, and decimal value 1. If the switch is already true, it sets the global switch to false and calls the same Bluetooth function with signed false and decimal value 2.

Εικόνα 31 Διακόπτες για τον έλεγχο των χρωμάτων

The image shows two Scratch code blocks for LED control. The left block is for the 'LedOn' button and the right block is for the 'LedOff' button. Both start with an 'initialize global' block setting 'LedOnPressed' to false. The main logic is in a 'when Clicked' event block. The 'LedOn' block sets the background color of all LED components (LedTop, LedLeftBottom, LedLeftTop, LedBottom, LedRightBottom, LedRightTop) to green and sets 'global LedOnPressed' to true. It then calls 'BluetoothLE1 - WriteBytesWithResponse' with service UUID, characteristic UUID, signed false, and decimal value 1. The 'LedOff' block sets the background color of all LED components to grey and sets 'global LedOnPressed' to false. It then calls 'BluetoothLE1 - WriteBytesWithResponse' with service UUID, characteristic UUID, signed true, and decimal value 1.

Εικόνα 32 Κομπιά για ενεργοποίηση απενεργοποίηση όλων των Led

Βασικός Κώδικας Arduino

Στο Παράρτημα βρίσκεται ο κώδικας του Arduino για την αναγνώριση φωνητικών εντολών, παρακάτω βρίσκεται ο κώδικας της βασικής εφαρμογής του κώδικα και η αλληλεπίδρασή της με την εφαρμογή.

Βιβλιοθήκες

Εισαγωγή βιβλιοθηκών με σκοπό την παροχή επιπλέον λειτουργιών για το Arduino

```
// LIBRARIES
#include <ArduinoBLE.h>
#include <Arduino_HTS221.h>
#include <Arduino_LPS22HB.h>
#include <Arduino_APDS9960.h>
#include "Arduino_BMI270_BMM150.h"
#include <Adafruit_NeoPixel.h>
#include <math.h>
#include <PDM.h>
```

Χαρακτηριστικά Bluetooth

Ορισμός χαρακτηριστικών και υπηρεσιών Bluetooth Low Energy (BLE) χρησιμοποιώντας τη βιβλιοθήκη Arduino Ble. Αυτά τα χαρακτηριστικά και υπηρεσίες καθορίζουν τη διεπαφή επικοινωνίας για το γάντι και περιλαμβάνουν χαρακτηριστικά θερμοκρασίας, πίεσης, υγρασίας, απόστασης, κίνησης και έλεγχος των LED, μεταξύ άλλων.

```
BLEService SmartGlove("20B10010-E8F2-537E-4F6C-D104768A1214");
// Bluetooth® Low Energy LED Service

// Bluetooth® Low Energy LED Switch Characteristic - custom 128-bit UUID,
read and writable by central
BLEByteCharacteristic temperatureCharacteristic("19B10001-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic pressureCharacteristic("19B10002-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic humidityCharacteristic("19B10003-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic proximityCharacteristic("19B10004-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
```

```

BLEByteCharacteristic gestureCharacteristic("19B10005-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic redCharacteristic("19B10006-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic greenCharacteristic("19B10007-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic blueCharacteristic("19B10008-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledOnCharacteristic("19B10009-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledOffCharacteristic("19B10010-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledTopCharacteristic("19B10011-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledBottomCharacteristic("19B10012-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledLeftTopCharacteristic("19B10013-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledLeftBottomCharacteristic("19B10014-E8F2-537E-
4F6C-D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledRightTopCharacteristic("19B10015-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic ledRightBottomCharacteristic("19B10016-E8F2-537E-
4F6C-D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic fingerCharacteristic("19B10017-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic wristCharacteristic("19B10018-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic redLedCharacteristic("19B10019-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic greenLedCharacteristic("19B10020-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic blueLedCharacteristic("19B10021-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);
BLEByteCharacteristic soundCharacteristic("19B10022-E8F2-537E-4F6C-
D104768A1214", BLERead | BLEWrite | BLENotify);

```

Μεταβλητές

Ορισμός και αρχικοποίηση μεταβλητών, συμπεριλαμβανομένου του πίνακα sampleBuffer για το δειγματοληψία ήχου, το pi για την αποθήκευση της τιμής του π, διάφορες μεταβλητές για γωνίες και pitch/roll, τις μεταβλητές LED_PIN και LED_COUNT για τον έλεγχο των LED της οθόνης NeoPixel, το fingerPin και τον πίνακα finger για τον αισθητήρα των δακτύλων, τιμές

κατωφλίου, διάφορες σημαίες για την κατάσταση των LED και ελέγχου των αισθητήρων, καθώς και μεταβλητές για την αποθήκευση τιμών χρωμάτων και απόστασης.

```
// default number of output channels
static const char channels = 1;
// default PCM output frequency
static const int frequency = 16000;
// Buffer to read samples into, each sample is 16-bits
short sampleBuffer[512];
// Number of audio samples read
volatile int samplesRead;

double pi = 3.14159265359;
double x_angle, y_angle, z_angle;
double pitch, roll;
double resultRoll, resultPitch;

#define LED_PIN 5

// How many NeoPixels are attached to the Arduino?
#define LED_COUNT 6

// Declare our NeoPixel strip object:
Adafruit_NeoPixel pixels(LED_COUNT, LED_PIN, NEO_GRB + NEO_KHZ800);

int fingerPin[4] = { A0, A1, A2, A3 };
int finger[4];
int thresh[5] = { 50, 50, 50, 50, 50 };

unsigned long previousMillis;
unsigned long intervalMillis = 10000;

int red = 0;
int green = 150;
int blue = 0;

bool LedAllOn = false;
bool LedTopOn = false;
bool LedBottomOn = false;
bool LedLeftTopOn = false;
bool LedLeftBottomOn = false;
bool LedRightTopOn = false;
bool LedRightBottomOn = false;
```

```

bool checkFlat = false;
bool checkPitchUp = false;
bool checkPitchDown = false;
bool checkRollLeft = false;
bool checkRollRight = false;

bool checkLevel1 = false;
bool checkLevel2 = false;
bool checkLevel3 = false;
bool checkLevel4 = false;
bool checkLevel5 = false;

int proximity = 0;
int r = 0, g = 0, b = 0;
unsigned long lastUpdate = 0;

```

Set up

Στη συνάρτηση `setup()`, ο κώδικας αρχικοποιεί τη σειριακή επικοινωνία, ελέγχει την αρχικοποίηση των διάφορων αισθητήρων, αρχικοποιεί την ταινία LED NeoPixel, ρυθμίζει την υπηρεσία BLE και τα χαρακτηριστικά και ξεκινά την αναγγελία της υπηρεσίας Bluetooth.

```

void setup() {
  Serial.begin(9600);
  //while (!Serial);
  // begin initialization
  if (!BLE.begin()) {
    Serial.println("starting Bluetooth® Low Energy module failed!");

    while (1)
      ;
  }
  if (!APDS.begin()) {
    Serial.println("Error initializing APDS-9960 sensor.");
    while (true)
      ; // Stop forever
  }
  if (!HTS.begin()) {
    Serial.println("Failed to initialize humidity temperature sensor!");
  }
  if (!BARO.begin()) {
    Serial.println("Failed to initialize pressure sensor!");
  }
}

```

```

    while (1)
        ;
}

for (int i = 0; i < 4; i++) {
    pinMode(fingerPin[i], INPUT_PULLUP);
}

if (!IMU.begin()) {
    Serial.println("Failed to initialize IMU!");
    while (1)
        ;
}
Serial.print(IMU.accelerationSampleRate());
Serial.print(IMU.gyroscopeSampleRate());
Serial.print(IMU.magneticFieldSampleRate());

pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)
pixels.clear(); // Set all pixel colors to 'off'
pixels.show(); // Send the updated pixel colors to the hardware.

// set advertised local name and service UUID:
BLE.setLocalName("Smart_Glove");
BLE.setAdvertisedService(SmartGlove);

// add the characteristic to the service
SmartGlove.addCharacteristic(temperatureCharacteristic);
SmartGlove.addCharacteristic(pressureCharacteristic);
SmartGlove.addCharacteristic(humidityCharacteristic);
SmartGlove.addCharacteristic(proximityCharacteristic);
SmartGlove.addCharacteristic(gestureCharacteristic);
SmartGlove.addCharacteristic(redCharacteristic);
SmartGlove.addCharacteristic(greenCharacteristic);
SmartGlove.addCharacteristic(blueCharacteristic);
SmartGlove.addCharacteristic(ledOnCharacteristic);
SmartGlove.addCharacteristic(ledOffCharacteristic);
SmartGlove.addCharacteristic(ledTopCharacteristic);
SmartGlove.addCharacteristic(ledBottomCharacteristic);
SmartGlove.addCharacteristic(ledLeftTopCharacteristic);
SmartGlove.addCharacteristic(ledLeftBottomCharacteristic);
SmartGlove.addCharacteristic(ledRightTopCharacteristic);
SmartGlove.addCharacteristic(ledRightBottomCharacteristic);
SmartGlove.addCharacteristic(fingerCharacteristic);
SmartGlove.addCharacteristic(wristCharacteristic);
SmartGlove.addCharacteristic(redLedCharacteristic);

```

```

SmartGlove.addCharacteristic(greenLedCharacteristic);
SmartGlove.addCharacteristic(blueLedCharacteristic);
SmartGlove.addCharacteristic(soundCharacteristic);

// add service
BLE.addService(SmartGlove);

// set the initial value for the characeristic:
temperatureCharacteristic.writeValue(0);
pressureCharacteristic.writeValue(0);
humidityCharacteristic.writeValue(0);
proximityCharacteristic.writeValue(0);
gestureCharacteristic.writeValue(0);
redCharacteristic.writeValue(0);
greenCharacteristic.writeValue(0);
blueCharacteristic.writeValue(0);
ledOnCharacteristic.writeValue(0);
ledOffCharacteristic.writeValue(0);
ledTopCharacteristic.writeValue(0);
ledBottomCharacteristic.writeValue(0);
ledLeftTopCharacteristic.writeValue(0);
ledLeftBottomCharacteristic.writeValue(0);
ledRightTopCharacteristic.writeValue(0);
ledRightBottomCharacteristic.writeValue(0);
fingerCharacteristic.writeValue(0);
wristCharacteristic.writeValue(0);
redLedCharacteristic.writeValue(0);
greenLedCharacteristic.writeValue(0);
blueLedCharacteristic.writeValue(0);
soundCharacteristic.writeValue(0);

// start advertising
BLE.advertise();

pixels.clear();
pixels.show();

PDM.onReceive(onPDMdata);

if (!PDM.begin(channels, frequency)) {
  Serial.println("Failed to start PDM!");
  while (1)
    ;
}
delay(5000);

```



```

while (true) {
  // Wait for samples to be read
  if (samplesRead) {

    // Print samples to the serial monitor or plotter
    for (int i = 0; i < samplesRead; i++) {
      if (channels == 2) {
        Serial.print("L:");
        Serial.print(sampleBuffer[i]);
        Serial.print(" R:");
        i++;
      }
      //Serial.println(sampleBuffer[i]);
      if (sampleBuffer[i] > 700) {
        Serial.println("Wake Up");
        goto bailout;
      }
    }
    // Clear the read count
    samplesRead = 0;
  }
}
bailout:
  Serial.println("Start");
  flashLed(0, 150, 0);
}

```

Βρόχος

Στη συνάρτηση loop(), ο κώδικας ελέγχει αν μία κεντρική συσκευή (Bluetooth περιφερειακή συσκευή) έχει συνδεθεί με το γάντι. Εάν υπάρχει σύνδεση με κεντρική συσκευή, εισέρχεται σε μια while loop όπου πραγματοποιεί διάφορες αναγνώσεις αισθητήρων και ενημερώνει τα αντίστοιχα χαρακτηριστικά. Ο κώδικας επίσης ελέγχει τις δειγματοληπτικές στιγμές ήχου PDM και πραγματοποιεί ενέργειες με βάση τα ληφθέντα δείγματα.

```

void loop() {
  // listen for Bluetooth Low Energy peripherals to connect:
  BLEDevice central = BLE.central();
  // if a central is connected to peripheral:
  if (central) {
    Serial.print("Connected to central: ");
    flashLed(0, 0, 150);
    unsigned long connectMillis = millis();

```

```

// print the central's MAC address:
Serial.println(central.address());
// while the central is still connected to peripheral:
while (central.connected()) {
  if (millis() - connectMillis > 5000) {
    if (millis() - previousMillis > intervalMillis) {
      environmentalSensor();
      previousMillis = millis();
    }
    lightSensor();
    tiltSensor();
    fingerSensor();
    mic();
    readValues();
  }
}
// when the central disconnects, print it out:
Serial.print(F("Disconnected from central: "));
Serial.println(central.address());
pixels.clear();
pixels.show();
}
}

```

Συνάρτηση Περιβαλλοντικών Αισθητήρων

Η συνάρτηση `environmentalSensor()` διαβάζει τη θερμοκρασία, την υγρασία και την πίεση από τους αντίστοιχους αισθητήρες και ενημερώνει τα αντίστοιχα χαρακτηριστικά BLE.

```

void environmentalSensor() {
  float temperature = BARO.readTemperature();
  float humidity = HTS.readHumidity();
  float pressure = BARO.readPressure();

  temperatureCharacteristic.writeValue((byte)temperature);
  humidityCharacteristic.writeValue((byte)humidity);
  pressureCharacteristic.writeValue((byte)pressure);
  // print each of the sensor values
  Serial.print("Temperature = ");
  Serial.print(temperature);
  Serial.print(" °C || ");
  Serial.print("Humidity    = ");
  Serial.print(humidity);
  Serial.print(" % || ");
  Serial.print("Pressure = ");

```

```

Serial.print(pressure);
Serial.print(" kPa || ");
}

```

Συνάρτηση Οπτικών Αισθητήρων

Η συνάρτηση lightSensor() διαβάζει τις τιμές απόστασης, κίνησης και χρώματος από τον αισθητήρα APDS9960 και ενημερώνει τα αντίστοιχα χαρακτηριστικά BLE.

```

void lightSensor() {
  // Check if a proximity reading is available.
  if (APDS.proximityAvailable()) {
    proximity = APDS.readProximity();
  }

  // Check if a gesture reading is available
  if (APDS.gestureAvailable()) {
    int gesture = APDS.readGesture();
    switch (gesture) {
      case GESTURE_UP:
        Serial.print("Detected UP gesture || ");
        gestureCharacteristic.writeValue((byte)1);
        break;

      case GESTURE_DOWN:
        Serial.print("Detected DOWN gesture || ");
        gestureCharacteristic.writeValue((byte)2);
        break;

      case GESTURE_LEFT:
        Serial.print("Detected LEFT gesture || ");
        gestureCharacteristic.writeValue((byte)3);
        break;

      case GESTURE_RIGHT:
        Serial.print("Detected RIGHT gesture ||");
        gestureCharacteristic.writeValue((byte)4);
        break;
      default:
        break;
    }
  }
}

```

```

// Check if a color reading is available
if (APDS.colorAvailable()) {
    APDS.readColor(r, g, b);
}

// Print updates every 100 ms
if (millis() - lastUpdate > 250) {
    lastUpdate = millis();
    proximityCharacteristic.writeValue((byte)proximity);
    redCharacteristic.writeValue((byte)r);
    greenCharacteristic.writeValue((byte)g);
    blueCharacteristic.writeValue((byte)b);
    Serial.print("PR=");
    Serial.print(proximity);
    Serial.print(" || RGB=");
    Serial.print(r);
    Serial.print(",");
    Serial.print(g);
    Serial.print(",");
    Serial.println(b);
}
}

```

Συνάρτηση Αισθητήρα Κλίσης

Η συνάρτηση `tiltSensor()` διαβάζει τις τιμές επιτάχυνσης από τον αισθητήρα IMU (Inertial Measurement Unit) και υπολογίζει γωνίες pitch και roll. Ενημερώνει τα αντίστοιχα χαρακτηριστικά BLE.

```

void tiltSensor() {
    float x, y, z;

    if (IMU.accelerationAvailable()) {
        IMU.readAcceleration(y, x, z);

        resultPitch = sqrt((y * y) + (z * z));
        resultPitch = x / resultPitch;
        x_angle = atan(resultPitch);
        roll = (x_angle * 180) / pi;

        resultRoll = sqrt((x * x) + (z * z));
        resultRoll = y / resultRoll;
        y_angle = atan(resultRoll);
        pitch = ((y_angle * 180) / pi);
    }
}

```

```

if (pitch < -35 && checkPitchUp == false) {
    Serial.print("Up ||");
    wristCharacteristic.writeValue((byte)1);
    checkFlat = false;
    checkPitchUp = true;
    checkPitchDown = false;
    checkRollLeft = false;
    checkRollRight = false;
} else if (pitch > 20 && checkPitchDown == false) {
    Serial.print("Down ||");
    wristCharacteristic.writeValue((byte)2);
    checkFlat = false;
    checkPitchUp = false;
    checkPitchDown = true;
    checkRollLeft = false;
    checkRollRight = false;
} else if (roll < -20 && checkRollLeft == false) {
    Serial.print("Left Roll ||");
    wristCharacteristic.writeValue((byte)3);
    checkFlat = false;
    checkPitchUp = false;
    checkPitchDown = false;
    checkRollLeft = true;
    checkRollRight = false;
} else if (roll > 20 && checkRollRight == false) {
    Serial.print("Right Roll ||");
    wristCharacteristic.writeValue((byte)4);
    checkFlat = false;
    checkPitchUp = false;
    checkPitchDown = false;
    checkRollLeft = false;
    checkRollRight = true;

} else if(pitch > -35 && pitch < 20 && roll > -20 && roll < 20 &&
checkFlat == false) {
    Serial.print("Flat ||");
    wristCharacteristic.writeValue((byte)5);
    checkFlat = true;
    checkPitchUp = false;
    checkPitchDown = false;
    checkRollLeft = false;
    checkRollRight = false;
}
}
/*

```

```

Serial.print("Pitch = ");
Serial.print(pitch);
Serial.print("Deg ||");
Serial.print(" Roll = ");
Serial.print(roll);
Serial.println("Deg");
*/
}

```

Συνάρτηση Αισθητήρων Δακτύλων

Η συνάρτηση `fingersensor()` ανιχνεύει την πίεση στα δάχτυλα, όπου γίνεται ανάγνωση των τιμών από αναλογικούς αισθητήρες για κάθε δάκτυλο. Ενημερώνει τα αντίστοιχα χαρακτηριστικά BLE.

```

void fingerSensor() {
  for (int i = 0; i < 4; i++) {
    finger[i] = analogRead(fingerPin[i]);
    switch (finger[i]) {
      case 0:
        Serial.print("Pressed! finger Num: ");
        Serial.print(i);
        Serial.print(" || ");
        fingerCharacteristic.writeValue((byte)i);
        break;
      default:
        //Serial.print("No Pressed! finger Num: ");
        //fingerCharacteristic.writeValue((byte)6);
        //Serial.println(i);
        break;
    }
    //Serial.print("finger ");
    //Serial.print(i);
    //Serial.print(": ");
    //Serial.println(finger[i]);
  }
  if (finger[0] == 0 && finger[1] == 0 && finger[2] == 0 && finger[3] ==
0) {
    fingerCharacteristic.writeValue((byte)5);
  }
}

```

Συνάρτηση Αισθητήρων Ήχου

Ο παρακάτω κώδικας επιτρέπει την ανάγνωση των δεδομένων ήχου από τον πυκνωτή (buffer) και την κατηγοριοποίησή τους σε διάφορα επίπεδα έντασης ήχου. Η συνάρτηση onPDMdata() εκτελείται όταν υπάρχουν δεδομένα ήχου διαθέσιμα για ανάγνωση. Αρχικά, η μεταβλητή bytesAvailable παίρνει τον αριθμό των διαθέσιμων bytes για ανάγνωση από το PDM. Στη συνέχεια, τα δεδομένα αναγνωρίζονται από τον πυκνωτή και αποθηκεύονται στον πίνακα sampleBuffer. Επειδή τα δεδομένα αναπαριστούν 16-bit δείγματα ήχου (2 bytes ανά δείγμα), η μεταβλητή samplesRead παίρνει την τιμή του bytesAvailable δια το 2, προκειμένου να υπολογιστεί ο αριθμός των δειγμάτων ήχου που διαβάστηκαν. Η συνάρτηση mic() ελέγχει τα δεδομένα ήχου που διαβάστηκαν και τα κατηγοριοποιεί σε διάφορα επίπεδα έντασης ήχου.

```
void onPDMdata() {
    // Query the number of available bytes
    int bytesAvailable = PDM.available();

    // Read into the sample buffer
    PDM.read(sampleBuffer, bytesAvailable);

    // 16-bit, 2 bytes per sample
    samplesRead = bytesAvailable / 2;
}
void mic() {
    if (samplesRead) {

        // Print samples to the serial monitor or plotter
        for (int i = 0; i < samplesRead; i++) {
            //Serial.println(sampleBuffer[i]);
            if (sampleBuffer[i] > 1500 && checkLevel5 == false) {
                Serial.println("Sound Level 1");
                checkLevel1 = false;
                checkLevel2 = false;
                checkLevel3 = false;
                checkLevel4 = false;
                checkLevel5 = true;
                soundCharacteristic.writeValue((byte)5);
            } else if (sampleBuffer[i] > 1200 && sampleBuffer[i] < 1500 &&
checkLevel4 == false) {
                Serial.println("Sound Level 4");
                checkLevel1 = false;
                checkLevel2 = false;
                checkLevel3 = false;
                checkLevel4 = true;
                checkLevel5 = false;
                soundCharacteristic.writeValue((byte)4);
            }
        }
    }
}
```

```

    } else if (sampleBuffer[i] > 900 && sampleBuffer[i] < 1200 &&
checkLevel3 == false) {
        Serial.println("Sound Level 3");
        checkLevel1 = false;
        checkLevel2 = false;
        checkLevel3 = true;
        checkLevel4 = false;
        checkLevel5 = false;
        soundCharacteristic.writeValue((byte)3);
    } else if (sampleBuffer[i] > 600 && sampleBuffer[i] < 900 &&
checkLevel2 == false) {
        Serial.println("Sound Level 2");
        checkLevel1 = false;
        checkLevel2 = true;
        checkLevel3 = false;
        checkLevel4 = false;
        checkLevel5 = false;
        soundCharacteristic.writeValue((byte)2);
    } else if (sampleBuffer[i] < 600 && checkLevel1 == false) {
        Serial.println("Sound Level 1");
        checkLevel1 = true;
        checkLevel2 = false;
        checkLevel3 = false;
        checkLevel4 = false;
        checkLevel5 = false;
        soundCharacteristic.writeValue((byte)1);
    }
}
// Clear the read count
samplesRead = 0;
}
}

```

Συνάρτηση Διαβάσματος τιμών εισόδου από εφαρμογή

Η συνάρτηση `readValues()` αναλαμβάνει να διαβάσει τις τιμές διάφορων χαρακτηριστικών που προέρχονται από μια εφαρμογή και να εκτελέσει τις αντίστοιχες ενέργειες. Εάν ένα χαρακτηριστικό έχει εγγραφεί, τότε διαβάζεται η τιμή του και ανάλογα με αυτήν, ενεργοποιούνται ή απενεργοποιούνται οι αντίστοιχοι φωτεινοί δείκτες. Για παράδειγμα, εάν το χαρακτηριστικό `ledOnCharacteristic` έχει εγγραφεί και η τιμή είναι `0x01`, τότε όλοι οι φωτεινοί δείκτες ενεργοποιούνται και ρυθμίζεται η αντίστοιχη τιμή χρώματος (`red`, `green`, `blue`). Αντίστοιχα, αν το χαρακτηριστικό `ledOffCharacteristic` έχει εγγραφεί και η τιμή είναι `0x01`, τότε απενεργοποιούνται όλοι οι φωτεινοί δείκτες.


```

void readValues() {

    if (redLedCharacteristic.written()) {
        // yes, get the value, characteristic is 1 byte so use byte value
        byte value = 0;
        redLedCharacteristic.readValue(value);
        if (value & 0x01) {
            red = 150;
            Serial.print("Red Value = 150 || ");
            updateLedColor();
        }
        if (value & 0x02) {
            red = 0;
            Serial.print("Red Value = 0 ||");
            updateLedColor();
        }
    }

    if (greenLedCharacteristic.written()) {
        // yes, get the value, characteristic is 1 byte so use byte value
        byte value = 0;
        greenLedCharacteristic.readValue(value);
        if (value & 0x01) {
            green = 150;
            Serial.print("Green Value = 150 || ");
            updateLedColor();
        }
        if (value & 0x02) {
            green = 0;
            Serial.print("Green Value = 0 || ");
            updateLedColor();
        }
    }

    if (blueLedCharacteristic.written()) {
        // yes, get the value, characteristic is 1 byte so use byte value
        byte value = 0;
        blueLedCharacteristic.readValue(value);
        if (value & 0x01) {
            blue = 150;
            Serial.print("Blue Value = 150 || ");
            updateLedColor();
        }
    }
}

```

```

    if (value & 0x02) {
        blue = 0;
        Serial.println("Blue Value = 0 || ");
        updateLedColor();
    }
}

if (ledOnCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledOnCharacteristic.readValue(value);
    if (value & 0x01) {
        for (int i = 0; i < 6; i++) {
            pixels.setPixelColor(i, pixels.Color(red, green, blue));
            pixels.show();
        }
        LedAllOn = true;
        LedTopOn = false;
        LedBottomOn = false;
        LedLeftTopOn = false;
        LedLeftBottomOn = false;
        LedRightTopOn = false;
        LedRightBottomOn = false;
        Serial.print("All On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
        Serial.print(blue);
        Serial.print(" || ");
    }
}

if (ledOffCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledOffCharacteristic.readValue(value);
    if (value & 0x01) {
        pixels.clear();
        pixels.show();
        Serial.print("All Off || ");
        LedAllOn = false;
        LedTopOn = false;
        LedBottomOn = false;
        LedLeftTopOn = false;
    }
}

```

```

    LedLeftBottomOn = false;
    LedRightTopOn = false;
    LedRightBottomOn = false;
}
}

if (ledTopCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledTopCharacteristic.readValue(value);
    if (value & 0x01) {
        LedTopOn = true;
        LedAllOn = false;
        pixels.setPixelColor(0, pixels.Color(red, green, blue));
        pixels.show();
        Serial.print("Top On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
        Serial.print(blue);
        Serial.print(" || ");
    }
    if (value & 0x02) {
        pixels.setPixelColor(0, pixels.Color(0, 0, 0));
        pixels.show();
        LedTopOn = false;
        Serial.print("Top Off || ");
    }
}

if (ledBottomCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledBottomCharacteristic.readValue(value);
    if (value & 0x01) {
        LedBottomOn = true;
        LedAllOn = false;
        pixels.setPixelColor(3, pixels.Color(red, green, blue));
        pixels.show();
        Serial.print("Bottom On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
        Serial.print(blue);
    }
}

```

```

        Serial.print(blue);
        Serial.print(" || ");
    }
    if (value & 0x02) {
        LedBottomOn = false;
        pixels.setPixelColor(3, pixels.Color(0, 0, 0));
        pixels.show();
        Serial.print("Bottom Off || ");
    }
}

if (ledLeftTopCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledLeftTopCharacteristic.readValue(value);
    if (value & 0x01) {
        LedLeftTopOn = true;
        LedAllOn = false;
        pixels.setPixelColor(5, pixels.Color(red, green, blue));
        pixels.show();
        Serial.print("Left Top On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
        Serial.print(blue);
        Serial.print(" || ");
    }
    if (value & 0x02) {
        LedLeftTopOn = false;
        pixels.setPixelColor(5, pixels.Color(0, 0, 0));
        pixels.show();
        Serial.print("Left Top Off || ");
    }
}

if (ledLeftBottomCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledLeftBottomCharacteristic.readValue(value);
    if (value & 0x01) {
        LedLeftBottomOn = true;
        LedAllOn = false;
        pixels.setPixelColor(4, pixels.Color(red, green, blue));
        pixels.show();
    }
}

```

```

    Serial.print("Left Bottom On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (value & 0x02) {
    LedLeftBottomOn = false;
    pixels.setPixelColor(4, pixels.Color(0, 0, 0));
    pixels.show();
    Serial.print("Left Bottom Off || ");
}
}

if (ledRightTopCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledRightTopCharacteristic.readValue(value);
    if (value & 0x01) {
        LedRightTopOn = true;
        LedAllOn = false;
        pixels.setPixelColor(1, pixels.Color(red, green, blue));
        pixels.show();
        Serial.print("Right Top On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
        Serial.print(blue);
        Serial.print(" || ");
    }
    if (value & 0x02) {
        LedRightTopOn = false;
        pixels.setPixelColor(1, pixels.Color(0, 0, 0));
        pixels.show();
        Serial.print("Right Top Off || ");
    }
}

if (ledRightBottomCharacteristic.written()) {
    // yes, get the value, characteristic is 1 byte so use byte value
    byte value = 0;
    ledRightBottomCharacteristic.readValue(value);

```

```

if (value & 0x01) {
    LedRightBottomOn = true;
    LedAllOn = false;
    pixels.setPixelColor(2, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Right Bottom On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (value & 0x02) {
    LedRightBottomOn = false;
    pixels.setPixelColor(2, pixels.Color(0, 0, 0));
    pixels.show();
    Serial.print("Right Bottom Off || ");
}
}
}
}

```

Συνάρτηση λειτουργίας LED

Η συνάρτηση `updateLedColor()` ελέγχει τις μεταβλητές `LedAllOn`, `LedTopOn`, `LedBottomOn`, `LedLeftTopOn`, `LedLeftBottomOn`, `LedRightTopOn`, `LedRightBottomOn` και εάν κάποια από αυτές είναι `true`, τότε ενημερώνει το χρώμα των αντίστοιχων φωτεινών δεικτών. Αυτό γίνεται με τη χρήση της βιβλιοθήκης `pixels` και την κλήση της μεθόδου `setPixelColor()`. Επίσης, εκτυπώνονται μηνύματα που περιέχουν ποιο χρώμα που ενημερώθηκε.

Η συνάρτηση `flashLed()` αναβοσβήνει τους φωτεινούς δείκτες σε συγκεκριμένο χρώμα (που δίνεται ως όρισμα στη συνάρτηση) για διάφορα χρονικά διαστήματα, χρησιμοποιώντας τις μεθόδους `setPixelColor()`, `clear()`, `show()` και τη συνάρτηση `delay()`.

```

void updateLedColor() {
    if (LedAllOn == true) {
        for (int i = 0; i < 6; i++) {
            pixels.setPixelColor(i, pixels.Color(red, green, blue));
            pixels.show();
        }
        Serial.print("Update Color, All On, Color:");
        Serial.print(red);
        Serial.print(",");
        Serial.print(green);
        Serial.print(",");
    }
}

```

```

    Serial.print(blue);
    Serial.print(" || ");
}
if (LedTopOn == true) {
    pixels.setPixelColor(0, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Top On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (LedBottomOn == true) {
    pixels.setPixelColor(3, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Bottom On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (LedLeftTopOn == true) {
    pixels.setPixelColor(5, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Left Top On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (LedLeftBottomOn == true) {
    pixels.setPixelColor(4, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Left Bottom On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
}

```

```

    Serial.print(" || ");
}
if (LedRightTopOn == true) {
    pixels.setPixelColor(1, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Right Top On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
if (LedRightBottomOn == true) {
    pixels.setPixelColor(2, pixels.Color(red, green, blue));
    pixels.show();
    Serial.print("Update Color, Right Bottom On, Color:");
    Serial.print(red);
    Serial.print(",");
    Serial.print(green);
    Serial.print(",");
    Serial.print(blue);
    Serial.print(" || ");
}
}
}
void flashLed(int red, int green, int blue) {
    for (int i = 0; i < 6; i++) {
        pixels.setPixelColor(i, pixels.Color(red, green, blue));
        pixels.show();
        delay(200);
    }
    delay(200);
    pixels.clear();
    pixels.show();
    delay(200);
    for (int i = 0; i < 6; i++) {
        pixels.setPixelColor(i, pixels.Color(red, green, blue));
        pixels.show();
    }
    delay(200);
    pixels.clear();
    pixels.show();
    delay(200);
    for (int i = 0; i < 6; i++) {
        pixels.setPixelColor(i, pixels.Color(red, green, blue));

```



```
    pixels.show();  
  }  
  delay(200);  
  pixels.clear();  
  pixels.show();  
}
```

ΣΥΜΠΕΡΑΣΜΑΤΑ

Για τη λήψη εμπειρισταωμένων συμπερασμάτων διοργανώθηκε μία δοκιμή λειτουργίας. Οι χρήστες της έπρεπε να αξιολογήσουν την εφαρμογή του γαντιού, τη συνδεσιμότητα με την εφαρμογή, καθώς και τις πληροφορίες που εμφανίζονται από την εφαρμογή.

Κατόπιν της δοκιμής, ολόκληρο το project έλαβε θετικά σχόλια και αναγνωρίστηκαν οι δυνατότητες που υπάρχουν και η πληθώρα εφαρμογών που μπορεί να χρησιμοποιηθεί το γάντι. Οι χρήστες βρήκαν πολύ χρήσιμες τις πληροφορίες των περιβαλλοντικών συνθηκών για τον χώρο που χρησιμοποιούν τη συσκευή, καθώς και τον φωτισμό που είναι χρήσιμος για περιστάσεις χαμηλής φωτεινότητας. Οι φωνητικές εντολές συγκέντρωσαν πολύ θετικά σχόλια καθώς οι δυνατότητες είναι απεριόριστες σε συνδυασμό συνδεσιμότητα γαντιού-εφαρμογής.

Λαμβάνοντας υπόψιν τα σχόλια που έγιναν αλλά και τις γενικότερες παρατηρήσεις κατά τη διάρκεια της κατασκευής, οι προτάσεις για επέκταση των δυνατοτήτων είναι οι παρακάτω. Το βασικότερο κομμάτι των βελτιώσεων αφορά στην αύξηση των φωνητικών εντολών το οποίο συνδέεται και με την αύξηση της μνήμης ROM του ελεγκτή. Επιπλέον, έγιναν προτάσεις για την αξιοποίηση των πληροφοριών που λαμβάνονται από το γάντι, ώστε να δημιουργηθούν αυτοματοποιημένες διεργασίες ή ακόμα και να γίνεται πληκτρολόγηση κειμένου.

Όσον αφορά στο τεχνικό κομμάτι, θα μπορούσε σε επόμενο στάδιο να προστεθούν περισσότεροι αισθητήρες οι οποίοι θα έκαναν έλεγχο στην υγεία του χρήστη, π.χ. αισθητήρες καρδιακών παλμών, οξυγόνου, κ.α. οι οποίοι εμπλουτίζουν τις δυνατότητες του γαντιού και βοηθούν τον χρήστη.

ΒΙΒΛΙΟΓΡΑΦΙΑ

-
- [1] Arduino.cc, "Arduino Reference," [Online]. Available: <https://www.arduino.cc/reference/en/language/functions/digital-io/digitalread/>. [Accessed 18 06 2023].
-
- [2] Liu D., Zhang Y., Zhou H., "A Copenhensive Study of Bluetooth Low Energy," Journal of Physics Conference Series , 2023. Available: https://www.researchgate.net/publication/356241573_A_Copenhensive_Study_of_Bluetoth_Low_Energy [Accessed 20 06 2023].
-
- [3] Aggarwal et al, "Has the Future Started? The Current Growth of Artificial Intelligence, Machine Learning, and Deep Learning" JIraqi Journal for Computer Science and Mathematics, 2022. Available: <https://www.iasj.net/iasj/download/cefbfd60eb11898a> [Accessed 20 06 2023].
-
- [4] Joel Mathew Cherian, Ravindra Kumar, "Fundamentals of Machine Learning", 2023. Available: https://www.researchgate.net/publication/371765323_Fundamentals_of_Machine_Learning [Accessed 20 06 2023].
-
- [5] Richard F. Tinder, " Relativistic Flight Mechanics and Space Travel: A Primer for Students, Engineers and Scientists ", 2007. Available: https://books.google.gr/books?id=-uMRwLaNbC8C&pg=PA33&redir_esc=y#v=onepage&q&f=false [Accessed 20 06 2023].
-
- [6] Πογαρίδης Δ., " Ενσωματωμένα συστήματα" Αθήνα: Εκδόσεις Δισίγμα, 2015.
-
- [7] Ali Salehabadi, Morteza Enhessari, Mardiana Idayu Ahmad, Norli Ismail, Banshi Dhar Gupta, "Metal Chalcogenide Biosensors", 2023. Available:
-

<https://www.sciencedirect.com/science/article/abs/pii/B978032385381100009X> [Accessed 20 06 2023].

- [8] Docs.arduino.cc, "Nano 33 BLE Sense", [Online]. Available: https://docs.arduino.cc/hardware/nano-33-ble-sense?fbclid=IwAR3JJmtu_4swWtorqj0fPTWg_gjSPHuABvnbSwl_Viwt7ctza6M-0WK5ba0 [Accessed 29 06 2023].
- [9] Nordic, " nRF52840", [Online]. Available: https://infocenter.nordicsemi.com/index.jsp?topic=%2Fstruct_nrf52%2Fstruct%2Fnrf52840.html [Accessed 29 06 2023].
- [10] Tanguy Risset, " SoC (System on Chip)", 2011. Available: https://doi.org/10.1007/978-0-387-09766-4_5 [Accessed 29 06 2023].

ΠΑΡΑΡΤΗΜΑ

```
/* Copyright 2023 The TensorFlow Authors. All Rights Reserved.
```

```
Licensed under the Apache License, Version 2.0 (the "License");  
you may not use this file except in compliance with the License.  
You may obtain a copy of the License at
```

```
http://www.apache.org/licenses/LICENSE-2.0
```

```
Unless required by applicable law or agreed to in writing, software  
distributed under the License is distributed on an "AS IS" BASIS,  
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
See the License for the specific language governing permissions and  
limitations under the License.
```

```
=====  
====*/
```

```
#include <TensorFlowLite.h>
```

```
#include "audio_provider.h"
```

```
#include "command_responder.h"
```

```
#include "feature_provider.h"
```

```
#include "main_functions.h"
```

```
#include "micro_features_micro_model_settings.h"
```

```
#include "micro_features_model.h"
```

```
#include "recognize_commands.h"
```

```
#include "tensorflow/lite/micro/micro_interpreter.h"
```

```
#include "tensorflow/lite/micro/micro_log.h"
```

```
#include "tensorflow/lite/micro/micro_mutable_op_resolver.h"
```

```
#include "tensorflow/lite/micro/system_setup.h"
```

```
#include "tensorflow/lite/schema/schema_generated.h"
```

```
#undef PROFILE_MICRO_SPEECH
```

```
// Globals, used for compatibility with Arduino-style sketches.
```

```
namespace {
```

```
const tflite::Model* model = nullptr;
```

```
tflite::MicroInterpreter* interpreter = nullptr;
```

```
TfLiteTensor* model_input = nullptr;
```

```
FeatureProvider* feature_provider = nullptr;
```

```
RecognizeCommands* recognizer = nullptr;
```

```
int32_t previous_time = 0;
```

```

// Create an area of memory to use for input, output, and intermediate
arrays.
// The size of this will depend on the model you're using, and may need to
be
// determined by experimentation.
constexpr int kTensorArenaSize = 10 * 1024;
// Keep aligned to 16 bytes for CMSIS
alignas(16) uint8_t tensor_arena[kTensorArenaSize];
int8_t feature_buffer[kFeatureElementCount];
int8_t* model_input_buffer = nullptr;
} // namespace

// The name of this function is important for Arduino compatibility.
void setup() {
    tflite::InitializeTarget();

    // Map the model into a usable data structure. This doesn't involve any
    // copying or parsing, it's a very lightweight operation.
    model = tflite::GetModel(g_model);
    if (model->version() != TFLITE_SCHEMA_VERSION) {
        MicroPrintf(
            "Model provided is schema version %d not equal "
            "to supported version %d.",
            model->version(), TFLITE_SCHEMA_VERSION);
        return;
    }

    // Pull in only the operation implementations we need.
    // This relies on a complete list of all the ops needed by this graph.
    // An easier approach is to just use the AllOpsResolver, but this will
    // incur some penalty in code space for op implementations that are not
    // needed by this graph.
    //
    // tflite::AllOpsResolver resolver;
    // NOLINTNEXTLINE(runtime-global-variables)
    static tflite::MicroMutableOpResolver<4> micro_op_resolver;
    if (micro_op_resolver.AddDepthwiseConv2D() != kTfLiteOk) {
        return;
    }
    if (micro_op_resolver.AddFullyConnected() != kTfLiteOk) {
        return;
    }
    if (micro_op_resolver.AddSoftmax() != kTfLiteOk) {
        return;
    }
}

```

```

if (micro_op_resolver.AddReshape() != kTfLiteOk) {
    return;
}

// Build an interpreter to run the model with.
static tf::MicroInterpreter static_interpreter(
    model, micro_op_resolver, tensor_arena, kTensorArenaSize);
interpreter = &static_interpreter;

// Allocate memory from the tensor_arena for the model's tensors.
TfLiteStatus allocate_status = interpreter->AllocateTensors();
if (allocate_status != kTfLiteOk) {
    MicroPrintf("AllocateTensors() failed");
    return;
}

// Get information about the memory area to use for the model's input.
model_input = interpreter->input(0);
if ((model_input->dims->size != 2) || (model_input->dims->data[0] != 1)
||
    (model_input->dims->data[1] !=
     (kFeatureSliceCount * kFeatureSliceSize)) ||
    (model_input->type != kTfLiteInt8)) {
    MicroPrintf("Bad input tensor parameters in model");
    return;
}
model_input_buffer = model_input->data.int8;

// Prepare to access the audio spectrograms from a microphone or other
source
// that will provide the inputs to the neural network.
// NOLINTNEXTLINE(runtime-global-variables)
static FeatureProvider static_feature_provider(kFeatureElementCount,
                                              feature_buffer);

feature_provider = &static_feature_provider;

static RecognizeCommands static_recognizer;
recognizer = &static_recognizer;

previous_time = 0;

// start the audio
TfLiteStatus init_status = InitAudioRecording();
if (init_status != kTfLiteOk) {
    MicroPrintf("Unable to initialize audio");
}

```

```

    return;
}

    MicroPrintf("Initialization complete");
}

// The name of this function is important for Arduino compatibility.
void loop() {
#ifdef PROFILE_MICRO_SPEECH
    const uint32_t prof_start = millis();
    static uint32_t prof_count = 0;
    static uint32_t prof_sum = 0;
    static uint32_t prof_min = std::numeric_limits<uint32_t>::max();
    static uint32_t prof_max = 0;
#endif // PROFILE_MICRO_SPEECH

    // Fetch the spectrogram for the current time.
    const int32_t current_time = LatestAudioTimestamp();
    int how_many_new_slices = 0;
    TfLiteStatus feature_status = feature_provider->PopulateFeatureData(
        previous_time, current_time, &how_many_new_slices);
    if (feature_status != kTfLiteOk) {
        MicroPrintf("Feature generation failed");
        return;
    }
    previous_time += how_many_new_slices * kFeatureSliceStrideMs;
    // If no new audio samples have been received since last time, don't
    bother
    // running the network model.
    if (how_many_new_slices == 0) {
        return;
    }

    // Copy feature buffer to input tensor
    for (int i = 0; i < kFeatureElementCount; i++) {
        model_input_buffer[i] = feature_buffer[i];
    }

    // Run the model on the spectrogram input and make sure it succeeds.
    TfLiteStatus invoke_status = interpreter->Invoke();
    if (invoke_status != kTfLiteOk) {
        MicroPrintf("Invoke failed");
        return;
    }
}

```



```

// Obtain a pointer to the output tensor
TfLiteTensor* output = interpreter->output(0);
// Determine whether a command was recognized based on the output of
inference
const char* found_command = nullptr;
uint8_t score = 0;
bool is_new_command = false;
TfLiteStatus process_status = recognizer->ProcessLatestResults(
    output, current_time, &found_command, &score, &is_new_command);
if (process_status != kTfLiteOk) {
    MicroPrintf("RecognizeCommands::ProcessLatestResults() failed");
    return;
}
// Do something based on the recognized command. The default
implementation
// just prints to the error console, but you should replace this with
your
// own function for a real application.
RespondToCommand(current_time, found_command, score, is_new_command);

#ifdef PROFILE_MICRO_SPEECH
const uint32_t prof_end = millis();
if (++prof_count > 10) {
    uint32_t elapsed = prof_end - prof_start;
    prof_sum += elapsed;
    if (elapsed < prof_min) {
        prof_min = elapsed;
    }
    if (elapsed > prof_max) {
        prof_max = elapsed;
    }
    if (prof_count % 300 == 0) {
        MicroPrintf("## time: min %dms max %dms avg %dms", prof_min,
prof_max,
                    prof_sum / prof_count);
    }
}
#endif // PROFILE_MICRO_SPEECH
}

```

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

```
=====  
====*/
```

```
#if defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)  
#define ARDUINO_EXCLUDE_CODE  
#endif // defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)  
  
#ifndef ARDUINO_EXCLUDE_CODE  
  
#include <algorithm>  
#include <cmath>  
  
#include "PDM.h"  
#include "audio_provider.h"  
#include "micro_features_micro_model_settings.h"  
#include "tensorflow/lite/micro/micro_log.h"  
#include "test_over_serial/test_over_serial.h"  
  
using namespace test_over_serial;  
  
namespace {  
bool g_is_audio_initialized = false;  
// An internal buffer able to fit 16x our sample size  
constexpr int kAudioCaptureBufferSize = DEFAULT_PDM_BUFFER_SIZE * 16;  
int16_t g_audio_capture_buffer[kAudioCaptureBufferSize];  
// A buffer that holds our output  
int16_t g_audio_output_buffer[kMaxAudioSampleSize];  
// Mark as volatile so we can check in a while loop to see if  
// any samples have arrived yet.  
volatile int32_t g_latest_audio_timestamp = 0;  
// test_over_serial sample index  
uint32_t g_test_sample_index;  
// test_over_serial silence insertion flag  
bool g_test_insert_silence = true;  
} // namespace  
  
void CaptureSamples() {
```

```

// This is how many bytes of new data we have each time this is called
const int number_of_samples = DEFAULT_PDM_BUFFER_SIZE / 2;
// Calculate what timestamp the last audio sample represents
const int32_t time_in_ms =
    g_latest_audio_timestamp +
    (number_of_samples / (kAudioSampleFrequency / 1000));
// Determine the index, in the history of all samples, of the last
sample
const int32_t start_sample_offset =
    g_latest_audio_timestamp * (kAudioSampleFrequency / 1000);
// Determine the index of this sample in our ring buffer
const int capture_index = start_sample_offset % kAudioCaptureBufferSize;
// Read the data to the correct place in our buffer
int num_read =
    PDM.read(g_audio_capture_buffer + capture_index,
DEFAULT_PDM_BUFFER_SIZE);
if (num_read != DEFAULT_PDM_BUFFER_SIZE) {
    MicroPrintf("### short read (%d/%d) @%dms", num_read,
        DEFAULT_PDM_BUFFER_SIZE, time_in_ms);
    while (true) {
        // NORETURN
    }
}
// This is how we let the outside world know that new audio data has
arrived.
g_latest_audio_timestamp = time_in_ms;
}

TfLiteStatus InitAudioRecording() {
    if (!g_is_audio_initialized) {
        // Hook up the callback that will be called with each sample
        PDM.onReceive(CaptureSamples);
        // Start listening for audio: MONO @ 16KHz
        PDM.begin(1, kAudioSampleFrequency);
        // gain: -20db (min) + 6.5db (13) + 3.2db (builtin) = -10.3db
        PDM.setGain(13);
        // Block until we have our first audio sample
        while (!g_latest_audio_timestamp) {
        }
        g_is_audio_initialized = true;
    }

    return kTfLiteOk;
}

```

```

TfLiteStatus GetAudioSamples(int start_ms, int duration_ms,
                             int* audio_samples_size, int16_t**
audio_samples) {
    // This next part should only be called when the main thread notices
    that the
    // latest audio sample data timestamp has changed, so that there's new
    data
    // in the capture ring buffer. The ring buffer will eventually wrap
    around and
    // overwrite the data, but the assumption is that the main thread is
    checking
    // often enough and the buffer is large enough that this call will be
    made
    // before that happens.

    // Determine the index, in the history of all samples, of the first
    // sample we want
    const int start_offset = start_ms * (kAudioSampleFrequency / 1000);
    // Determine how many samples we want in total
    const int duration_sample_count =
        duration_ms * (kAudioSampleFrequency / 1000);
    for (int i = 0; i < duration_sample_count; ++i) {
        // For each sample, transform its index in the history of all samples
        into
        // its index in g_audio_capture_buffer
        const int capture_index = (start_offset + i) %
kAudioCaptureBufferSize;
        // Write the sample to the output buffer
        g_audio_output_buffer[i] = g_audio_capture_buffer[capture_index];
    }

    // Set pointers to provide access to the audio
    *audio_samples_size = duration_sample_count;
    *audio_samples = g_audio_output_buffer;

    return kTfLiteOk;
}

namespace {

void InsertSilence(const size_t len, int16_t value) {
    for (size_t i = 0; i < len; i++) {
        const size_t index = (g_test_sample_index + i) %
kAudioCaptureBufferSize;
        g_audio_capture_buffer[index] = value;
    }
}
}

```

```

    }
    g_test_sample_index += len;
}

int32_t ProcessTestInput(TestOverSerial& test) {
    constexpr size_t samples_16ms = ((kAudioSampleFrequency / 1000) * 16);

    InputHandler handler = [](const InputBuffer* const input) {
        if (0 == input->offset) {
            // don't insert silence
            g_test_insert_silence = false;
        }

        for (size_t i = 0; i < input->length; i++) {
            const size_t index = (g_test_sample_index + i) %
kAudioCaptureBufferSize;
            g_audio_capture_buffer[index] = input->data.int16[i];
        }
        g_test_sample_index += input->length;

        if (input->total == (input->offset + input->length)) {
            // allow silence insertion again
            g_test_insert_silence = true;
        }
        return true;
    };

    test.ProcessInput(&handler);

    if (g_test_insert_silence) {
        // add 16ms of silence just like the PDM interface
        InsertSilence(samples_16ms, 0);
    }

    // Round the timestamp to a multiple of 64ms,
    // This emulates the PDM interface during inference processing.
    g_latest_audio_timestamp = (g_test_sample_index / (samples_16ms * 4)) *
64;
    return g_latest_audio_timestamp;
}

} // namespace

int32_t LatestAudioTimestamp() {

```

```

    TestOverSerial& test =
TestOverSerial::Instance(kAUDIO_PCM_16KHZ_MONO_S16);
    if (!test.IsTestMode()) {
        // check serial port for test mode command
        test.ProcessInput(nullptr);
    }
    if (test.IsTestMode()) {
        if (g_is_audio_initialized) {
            // stop capture from hardware
            PDM.end();
            g_is_audio_initialized = false;
            g_test_sample_index =
                g_latest_audio_timestamp * (kAudioSampleFrequency / 1000);
        }
        return ProcessTestInput(test);
    } else {
        // CaptureSamples() updated the timestamp
        return g_latest_audio_timestamp;
    }
    // NOTREACHED
}

```

```
#endif // ARDUINO_EXCLUDE_CODE
```

```
/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.
```

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

```
=====
====*/
```

```

#if defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)
#define ARDUINO_EXCLUDE_CODE
#endif // defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)

```

```
#ifndef ARDUINO_EXCLUDE_CODE
```

```

#include "Arduino.h"
#include "command_responder.h"
#include "tensorflow/lite/micro/micro_log.h"

// Toggles the built-in LED every inference, and lights a colored LED
// depending
// on which word was detected.
void RespondToCommand(int32_t current_time, const char* found_command,
                     uint8_t score, bool is_new_command) {
    static bool is_initialized = false;
    if (!is_initialized) {
        pinMode(LED_BUILTIN, OUTPUT);
        // Pins for the built-in RGB LEDs on the Arduino Nano 33 BLE Sense
        pinMode(LED_R, OUTPUT);
        pinMode(LED_G, OUTPUT);
        pinMode(LED_B, OUTPUT);
        // Ensure the LED is off by default.
        // Note: The RGB LEDs on the Arduino Nano 33 BLE
        // Sense are on when the pin is LOW, off when HIGH.
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);
        is_initialized = true;
    }
    static int32_t last_command_time = 0;
    static int count = 0;

    if (is_new_command) {
        MicroPrintf("Heard %s (%d) @%dms", found_command, score,
current_time);
        // If we hear a command, light up the appropriate LED
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);

        if (found_command[0] == 'y') {
            digitalWrite(LED_G, LOW); // Green for yes
        } else if (found_command[0] == 'n') {
            digitalWrite(LED_R, LOW); // Red for no
        } else if (found_command[0] == 'u') {
            digitalWrite(LED_B, LOW); // Blue for unknown
        } else {
            // silence
        }
    }
}

```

```

    last_command_time = current_time;
}

// If last_command_time is non-zero but was >3 seconds ago, zero it
// and switch off the LED.
if (last_command_time != 0) {
    if (last_command_time < (current_time - 3000)) {
        last_command_time = 0;
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);
    }
}

// Otherwise, toggle the LED every time an inference is performed.
++count;
if (count & 1) {
    digitalWrite(LED_BUILTIN, HIGH);
} else {
    digitalWrite(LED_BUILTIN, LOW);
}
}

#endif // ARDUINO_EXCLUDE_CODE

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#if defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)
#define ARDUINO_EXCLUDE_CODE
#endif // defined(ARDUINO) && !defined(ARDUINO_ARDUINO_NANO33BLE)

```



```

#ifdef ARDUINO_EXCLUDE_CODE

#include "Arduino.h"
#include "command_responder.h"
#include "tensorflow/lite/micro/micro_log.h"

// Toggles the built-in LED every inference, and lights a colored LED
// depending
// on which word was detected.
void RespondToCommand(int32_t current_time, const char* found_command,
                      uint8_t score, bool is_new_command) {
    static bool is_initialized = false;
    if (!is_initialized) {
        pinMode(LED_BUILTIN, OUTPUT);
        // Pins for the built-in RGB LEDs on the Arduino Nano 33 BLE Sense
        pinMode(LED_R, OUTPUT);
        pinMode(LED_G, OUTPUT);
        pinMode(LED_B, OUTPUT);
        // Ensure the LED is off by default.
        // Note: The RGB LEDs on the Arduino Nano 33 BLE
        // Sense are on when the pin is LOW, off when HIGH.
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);
        is_initialized = true;
    }
    static int32_t last_command_time = 0;
    static int count = 0;

    if (is_new_command) {
        MicroPrintf("Heard %s (%d) @%dms", found_command, score,
current_time);
        // If we hear a command, light up the appropriate LED
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);

        if (found_command[0] == 'y') {
            digitalWrite(LED_G, LOW); // Green for yes
        } else if (found_command[0] == 'n') {
            digitalWrite(LED_R, LOW); // Red for no
        } else if (found_command[0] == 'u') {
            digitalWrite(LED_B, LOW); // Blue for unknown
        } else {

```

```

    // silence
}

last_command_time = current_time;
}

// If last_command_time is non-zero but was >3 seconds ago, zero it
// and switch off the LED.
if (last_command_time != 0) {
    if (last_command_time < (current_time - 3000)) {
        last_command_time = 0;
        digitalWrite(LED_R, HIGH);
        digitalWrite(LED_G, HIGH);
        digitalWrite(LED_B, HIGH);
    }
}

// Otherwise, toggle the LED every time an inference is performed.
++count;
if (count & 1) {
    digitalWrite(LED_BUILTIN, HIGH);
} else {
    digitalWrite(LED_BUILTIN, LOW);
}
}

#endif // ARDUINO_EXCLUDE_CODE

/* Copyright 2019 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#include "main_functions.h"

```

```

// Arduino automatically calls the setup() and loop() functions in a
// sketch, so
// where other systems need their own main routine in this file, it can be
// left
// empty.

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#ifdef TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_
#define TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_

#include "tensorflow/lite/c/common.h"

// This is an abstraction around an audio source like a microphone, and is
// expected to return 16-bit PCM sample data for a given point in time.
// The
// sample data itself should be used as quickly as possible by the caller,
// since
// to allow memory optimizations there are no guarantees that the samples
// won't
// be overwritten by new data in the future. In practice, implementations
// should
// ensure that there's a reasonable time allowed for clients to access the
// data
// before any reuse.
// The reference implementation can have no platform-specific
// dependencies, so
// it just returns an array filled with zeros. For real applications, you
// should

```

```

// ensure there's a specialized implementation that accesses hardware
APIs.
TfLiteStatus GetAudioSamples(int start_ms, int duration_ms,
                             int* audio_samples_size, int16_t**
audio_samples);

// Returns the time that audio data was last captured in milliseconds.
There's
// no contract about what time zero represents, the accuracy, or the
granularity
// of the result. Subsequent calls will generally not return a lower
value, but
// even that's not guaranteed if there's an overflow wraparound.
// The reference implementation of this function just returns a constantly
// incrementing value for each call, since it would need a non-portable
platform
// call to access time information. For real applications, you'll need to
write
// your own platform-specific implementation.
int32_t LatestAudioTimestamp();

// Starts audio capture
TfLiteStatus InitAudioRecording();

#endif // TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#ifndef TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_
#define TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_

```

```

#include "tensorflow/lite/c/common.h"

// This is an abstraction around an audio source like a microphone, and is
// expected to return 16-bit PCM sample data for a given point in time.
// The
// sample data itself should be used as quickly as possible by the caller,
// since
// to allow memory optimizations there are no guarantees that the samples
// won't
// be overwritten by new data in the future. In practice, implementations
// should
// ensure that there's a reasonable time allowed for clients to access the
// data
// before any reuse.
// The reference implementation can have no platform-specific
// dependencies, so
// it just returns an array filled with zeros. For real applications, you
// should
// ensure there's a specialized implementation that accesses hardware
// APIs.
TfLiteStatus GetAudioSamples(int start_ms, int duration_ms,
                             int* audio_samples_size, int16_t**
audio_samples);

// Returns the time that audio data was last captured in milliseconds.
// There's
// no contract about what time zero represents, the accuracy, or the
// granularity
// of the result. Subsequent calls will generally not return a lower
// value, but
// even that's not guaranteed if there's an overflow wraparound.
// The reference implementation of this function just returns a constantly
// incrementing value for each call, since it would need a non-portable
// platform
// call to access time information. For real applications, you'll need to
// write
// your own platform-specific implementation.
int32_t LatestAudioTimestamp();

// Starts audio capture
TfLiteStatus InitAudioRecording();

#endif // TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_AUDIO_PROVIDER_H_

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

```

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

```
=====
====*/
```

```
#include "feature_provider.h"
```

```
#include "audio_provider.h"
```

```
#include "micro_features_micro_features_generator.h"
```

```
#include "micro_features_micro_model_settings.h"
```

```
#include "tensorflow/lite/micro/micro_log.h"
```

```
FeatureProvider::FeatureProvider(int feature_size, int8_t* feature_data)
    : feature_size_(feature_size),
      feature_data_(feature_data),
      is_first_run_(true) {
    // Initialize the feature data to default values.
    for (int n = 0; n < feature_size_; ++n) {
        feature_data_[n] = 0;
    }
}
```

```
FeatureProvider::~FeatureProvider() {}
```

```
TfLiteStatus FeatureProvider::PopulateFeatureData(int32_t last_time_in_ms,
                                                  int32_t time_in_ms,
                                                  int*
```

```
how_many_new_slices) {
    if (feature_size_ != kFeatureElementCount) {
        MicroPrintf("Requested feature_data_ size %d doesn't match %d",
                    feature_size_, kFeatureElementCount);
        return kTfLiteError;
    }
}
```

```
// Quantize the time into steps as long as each window stride, so we can
```

```

// figure out which audio data we need to fetch.
const int last_step = (last_time_in_ms / kFeatureSliceStrideMs);
// Number of new 20ms slices from which we can take 30ms samples
int slices_needed =
    (((time_in_ms - last_time_in_ms) - kFeatureSliceDurationMs) *
     kFeatureSliceStrideMs) /
     kFeatureSliceStrideMs +
     kFeatureSliceStrideMs) /
     kFeatureSliceStrideMs;
// If this is the first call, make sure we don't use any cached
information.
if (is_first_run_) {
    TfLiteStatus init_status = InitializeMicroFeatures();
    if (init_status != kTfLiteOk) {
        return init_status;
    }
    is_first_run_ = false;
    return kTfLiteOk;
}
if (slices_needed > kFeatureSliceCount) {
    slices_needed = kFeatureSliceCount;
}
if (slices_needed == 0) {
    return kTfLiteOk;
}
*how_many_new_slices = slices_needed;

const int slices_to_keep = kFeatureSliceCount - slices_needed;
const int slices_to_drop = kFeatureSliceCount - slices_to_keep;
// If we can avoid recalculating some slices, just move the existing
data
// up in the spectrogram, to perform something like this:
// last time = 80ms          current time = 120ms
// +-----+                +-----+
// | data@20ms |            --> | data@60ms |
// +-----+                +-----+
// | data@40ms |          -- --> | data@80ms |
// +-----+          -- --    +-----+
// | data@60ms | -- --      | <empty> |
// +-----+          --      +-----+
// | data@80ms | --          | <empty> |
// +-----+                +-----+
if (slices_to_keep > 0) {
    for (int dest_slice = 0; dest_slice < slices_to_keep; ++dest_slice) {
        int8_t* dest_slice_data =

```

```

        feature_data_ + (dest_slice * kFeatureSliceSize);
const int src_slice = dest_slice + slices_to_drop;
const int8_t* src_slice_data =
    feature_data_ + (src_slice * kFeatureSliceSize);
for (int i = 0; i < kFeatureSliceSize; ++i) {
    dest_slice_data[i] = src_slice_data[i];
}
}
}
// Any slices that need to be filled in with feature data have their
// appropriate audio data pulled, and features calculated for that
slice.
if (slices_needed > 0) {
    for (int new_slice = slices_to_keep; new_slice < kFeatureSliceCount;
        ++new_slice) {
        const int new_step = last_step + (new_slice - slices_to_keep);
        const int32_t slice_start_ms = (new_step * kFeatureSliceStrideMs);
        int16_t* audio_samples = nullptr;
        int audio_samples_size = 0;
        GetAudioSamples(slice_start_ms, kFeatureSliceDurationMs,
            &audio_samples_size, &audio_samples);
        constexpr int wanted =
            kFeatureSliceDurationMs * (kAudioSampleFrequency / 1000);
        if (audio_samples_size != wanted) {
            MicroPrintf("Audio data size %d too small, want %d",
audio_samples_size,
                wanted);
            return kTfLiteError;
        }
        int8_t* new_slice_data = feature_data_ + (new_slice *
kFeatureSliceSize);
        size_t num_samples_read;
        TfLiteStatus generate_status = GenerateMicroFeatures(
            audio_samples, audio_samples_size, kFeatureSliceSize,
new_slice_data,
                &num_samples_read);
        if (generate_status != kTfLiteOk) {
            return generate_status;
        }
    }
}
return kTfLiteOk;
}

```

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

=====
====*/

```
#ifndef TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_FEATURE_PROVIDER_H_
#define TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_FEATURE_PROVIDER_H_
```

```
#include "tensorflow/lite/c/common.h"
```

```
// Binds itself to an area of memory intended to hold the input features  
for an  
// audio-recognition neural network model, and fills that data area with  
the  
// features representing the current audio input, for example from a  
microphone.
```

```
// The audio features themselves are a two-dimensional array, made up of  
// horizontal slices representing the frequencies at one point in time,  
stacked  
// on top of each other to form a spectrogram showing how those  
frequencies  
// changed over time.
```

```
class FeatureProvider {  
public:
```

```
    // Create the provider, and bind it to an area of memory. This memory  
should
```

```
    // remain accessible for the lifetime of the provider object, since  
subsequent
```

```
    // calls will fill it with feature data. The provider does no memory  
// management of this data.
```

```
    FeatureProvider(int feature_size, int8_t* feature_data);
```

```
    ~FeatureProvider();
```

```
    // Fills the feature data with information from audio inputs, and  
returns how
```

```

    // many feature slices were updated.
    TfLiteStatus PopulateFeatureData(int32_t last_time_in_ms, int32_t
time_in_ms,
                                int* how_many_new_slices);

private:
    int feature_size_;
    int8_t* feature_data_;
    // Make sure we don't try to use cached information if this is the first
call
    // into the provider.
    bool is_first_run_;
};

#endif // TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_FEATURE_PROVIDER_H_

/* Copyright 2019 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#ifndef TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MAIN_FUNCTIONS_H_
#define TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MAIN_FUNCTIONS_H_

// Expose a C friendly interface for main functions.
#ifdef __cplusplus
extern "C" {
#endif

// Initializes all data needed for the example. The name is important, and
needs
// to be setup() for Arduino compatibility.
void setup();

```

```

// Runs one iteration of data gathering and inference. This should be
called
// repeatedly from the application code. The name needs to be loop() for
Arduino
// compatibility.
void loop();

#ifdef __cplusplus
}
#endif

#endif // TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MAIN_FUNCTIONS_H_

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#include "micro_features_micro_features_generator.h"

#include <cmath>
#include <cstring>

#include "micro_features_micro_model_settings.h"
#include "tensorflow/lite/experimental/microfrontend/lib/frontend.h"
#include "tensorflow/lite/experimental/microfrontend/lib/frontend_util.h"
#include "tensorflow/lite/micro/micro_log.h"

// Configure FFT to output 16 bit fixed point.
#define FIXED_POINT 16

namespace {

FrontendState g_micro_features_state;

```

```

bool g_is_first_time = true;

} // namespace

TfLiteStatus InitializeMicroFeatures() {
  FrontendConfig config;
  config.window.size_ms = kFeatureSliceDurationMs;
  config.window.step_size_ms = kFeatureSliceStrideMs;
  config.noise_reduction.smoothing_bits = 10;
  config.filterbank.num_channels = kFeatureSliceSize;
  config.filterbank.lower_band_limit = 125.0;
  config.filterbank.upper_band_limit = 7500.0;
  config.noise_reduction.smoothing_bits = 10;
  config.noise_reduction.even_smoothing = 0.025;
  config.noise_reduction.odd_smoothing = 0.06;
  config.noise_reduction.min_signal_remaining = 0.05;
  config.pcan_gain_control.enable_pcan = 1;
  config.pcan_gain_control.strength = 0.95;
  config.pcan_gain_control.offset = 80.0;
  config.pcan_gain_control.gain_bits = 21;
  config.log_scale.enable_log = 1;
  config.log_scale.scale_shift = 6;
  if (!FrontendPopulateState(&config, &g_micro_features_state,
                             kAudioSampleFrequency)) {
    MicroPrintf("FrontendPopulateState() failed");
    return kTfLiteError;
  }
  g_is_first_time = true;
  return kTfLiteOk;
}

// This is not exposed in any header, and is only used for testing, to
// ensure
// that the state is correctly set up before generating results.
void SetMicroFeaturesNoiseEstimates(const uint32_t* estimate_presets) {
  for (int i = 0; i < g_micro_features_state.filterbank.num_channels; ++i)
  {
    g_micro_features_state.noise_reduction.estimate[i] =
estimate_presets[i];
  }
}

TfLiteStatus GenerateMicroFeatures(const int16_t* input, int input_size,
                                   int output_size, int8_t* output,
                                   size_t* num_samples_read) {

```

```

const int16_t* frontend_input;
if (g_is_first_time) {
    frontend_input = input;
    g_is_first_time = false;
} else {
    frontend_input = input;
}
FrontendOutput frontend_output = FrontendProcessSamples(
    &g_micro_features_state, frontend_input, input_size,
num_samples_read);

for (size_t i = 0; i < frontend_output.size; ++i) {
    // These scaling values are derived from those used in input_data.py
in the
    // training pipeline.
    // The feature pipeline outputs 16-bit signed integers in roughly a 0
to 670
    // range. In training, these are then arbitrarily divided by 25.6 to
get
    // float values in the rough range of 0.0 to 26.0. This scaling is
performed
    // for historical reasons, to match up with the output of other
feature
    // generators.
    // The process is then further complicated when we quantize the model.
This
    // means we have to scale the 0.0 to 26.0 real values to the -128 to
127
    // signed integer numbers.
    // All this means that to get matching values from our integer feature
    // output into the tensor input, we have to perform:
    // input = (((feature / 25.6) / 26.0) * 256) - 128
    // To simplify this and perform it in 32-bit integer math, we
rearrange to:
    // input = (feature * 256) / (25.6 * 26.0) - 128
    constexpr int32_t value_scale = 256;
    constexpr int32_t value_div = static_cast<int32_t>((25.6f * 26.0f) +
0.5f);
    int32_t value =
        ((frontend_output.values[i] * value_scale) + (value_div / 2)) /
        value_div;
    value -= 128;
    if (value < -128) {
        value = -128;
    }
}

```

```

    if (value > 127) {
        value = 127;
    }
    output[i] = value;
}

return kTfLiteOk;
}

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#ifdef
TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MICRO_FEATURES_MICRO_FEATURES_
GENERATOR_H_
#define
TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MICRO_FEATURES_MICRO_FEATURES_
GENERATOR_H_

#include "tensorflow/lite/c/common.h"

// Sets up any resources needed for the feature generation pipeline.
TfLiteStatus InitializeMicroFeatures();

// Converts audio sample data into a more compact form that's appropriate
for
// feeding into a neural network.
TfLiteStatus GenerateMicroFeatures(const int16_t* input, int input_size,
                                   int output_size, int8_t* output,
                                   size_t* num_samples_read);

```

```

#endif //
tensorflow_lite_micro_examples_micro_speech_micro_features_micro_features_
generator_h_

/* Copyright 2018 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#include "micro_features_micro_model_settings.h"

const char* kCategoryLabels[kCategoryCount] = {
    "silence",
    "unknown",
    "yes",
    "no",
};

/* Copyright 2018 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

```

```

#include "micro_features_micro_model_settings.h"

const char* kCategoryLabels[kCategoryCount] = {
    "silence",
    "unknown",
    "yes",
    "no",
};

/* Copyright 2020 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#ifdef
TensorFlow_Lite_Micro_Examples_Micro_Speech_Micro_Features_Micro_Model_Set
tings_H_
#define
TensorFlow_Lite_Micro_Examples_Micro_Speech_Micro_Features_Micro_Model_Set
tings_H_

// Keeping these as constant expressions allow us to allocate fixed-sized
arrays
// on the stack for our working memory.

// The size of the input time series data we pass to the FFT to produce
the
// frequency information. This has to be a power of two, and since we're
dealing
// with 30ms of 16KHz inputs, which means 480 samples, this is the next
value.
constexpr int kMaxAudioSampleSize = 512;
constexpr int kAudioSampleFrequency = 16000;

```



```

// The following values are derived from values used during model
training.
// If you change the way you preprocess the input, update all these
constants.
constexpr int kFeatureSliceSize = 40;
constexpr int kFeatureSliceCount = 49;
constexpr int kFeatureElementCount = (kFeatureSliceSize *
kFeatureSliceCount);
constexpr int kFeatureSliceStrideMs = 20;
constexpr int kFeatureSliceDurationMs = 30;

// Variables for the model's output categories.
constexpr int kSilenceIndex = 0;
constexpr int kUnknownIndex = 1;
// If you modify the output categories, you need to update the following
values.
constexpr int kCategoryCount = 4;
extern const char* kCategoryLabels[kCategoryCount];

#endif //
TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_MICRO_FEATURES_MICRO_MODEL_SET
TINGS_H_

```

/* Copyright 2023 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

```
=====
=====*/
```

```
// This is a standard TensorFlow Lite FlatBuffer model file that has been
// converted into a C data array, so it can be easily compiled into a binary
// for devices that don't have a file system. It was created using the command:
// xxd -i model.tflite > model.cc
```

```
#include "micro_features_model.h"
```

```
// Keep aligned to 16 bytes for CMSIS
```

```
alignas(16) const unsigned char g_model[] = {
    0x20, 0x00, 0x00, 0x00, 0x54, 0x46, 0x4c, 0x33, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x12, 0x00, 0x1c, 0x00, 0x04, 0x00, 0x08, 0x00, 0x0c, 0x00,
    0x10, 0x00, 0x14, 0x00, 0x00, 0x00, 0x18, 0x00, 0x12, 0x00, 0x00, 0x00,
    0x03, 0x00, 0x00, 0x00, 0x94, 0x48, 0x00, 0x00, 0x34, 0x42, 0x00, 0x00,
    0x1c, 0x42, 0x00, 0x00, 0x3c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
    0x01, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00,
    0x04, 0x00, 0x08, 0x00, 0x08, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00,
    0x0b, 0x00, 0x00, 0x00, 0x13, 0x00, 0x00, 0x00, 0x6d, 0x69, 0x6e, 0x5f,
    0x72, 0x75, 0x6e, 0x74, 0x69, 0x6d, 0x65, 0x5f, 0x76, 0x65, 0x72, 0x73,
    0x69, 0x6f, 0x6e, 0x00, 0x0c, 0x00, 0x00, 0x00, 0xd4, 0x41, 0x00, 0x00,
    0xb4, 0x41, 0x00, 0x00, 0x24, 0x03, 0x00, 0x00, 0xf4, 0x02, 0x00, 0x00,
    0xec, 0x02, 0x00, 0x00, 0xe4, 0x02, 0x00, 0x00, 0xc4, 0x02, 0x00, 0x00,
    0xbc, 0x02, 0x00, 0x00, 0x2c, 0x00, 0x00, 0x00, 0x24, 0x00, 0x00, 0x00,
    0x1c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x16, 0xbd, 0xff, 0xff,
    0x04, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x31, 0x2e, 0x35, 0x2e,
```

0x30, 0x00, 0x00, 0x00, 0x94, 0xba, 0xff, 0xff, 0x98, 0xba, 0xff, 0xff,
0x32, 0xbd, 0xff, 0xff, 0x04, 0x00, 0x00, 0x00, 0x80, 0x02, 0x00, 0x00,
0xfa, 0xee, 0x28, 0xc4, 0xee, 0xfe, 0xcf, 0x0f, 0x1e, 0xf7, 0x1f, 0x06,
0x0d, 0xed, 0xe9, 0x83, 0x5c, 0xc9, 0x18, 0xe3, 0xf9, 0x14, 0x28, 0x2a,
0x09, 0xf2, 0x18, 0x34, 0x62, 0xea, 0xef, 0xd6, 0x36, 0xb7, 0x1e, 0xf7,
0x3b, 0x22, 0x28, 0x39, 0xc2, 0x9d, 0xf1, 0x07, 0x5e, 0x0b, 0x1e, 0x2c,
0x07, 0xdd, 0xfd, 0xc3, 0xd8, 0x4a, 0xf3, 0x28, 0xa7, 0x16, 0xd5, 0xf1,
0xc3, 0x05, 0xfd, 0x27, 0xcc, 0xba, 0x1e, 0xcb, 0xd7, 0x3d, 0xd4, 0x29,
0x00, 0xfd, 0x28, 0x44, 0xfb, 0xf2, 0xf3, 0xb6, 0x4f, 0xcf, 0x09, 0xf0,
0xfa, 0x45, 0x41, 0x49, 0x05, 0xc5, 0x17, 0x5d, 0x64, 0x00, 0xf8, 0xee,
0x48, 0x17, 0xf4, 0xe9, 0x2e, 0x4b, 0x2e, 0x3f, 0xdf, 0xee, 0xe4, 0x08,
0x38, 0xf1, 0x16, 0x13, 0x2f, 0x2a, 0xed, 0xc2, 0xbf, 0x36, 0xf4, 0x02,
0xcf, 0xaa, 0xd2, 0xfa, 0xac, 0x13, 0xf6, 0xe8, 0xb5, 0x68, 0x12, 0xb6,
0xce, 0x0e, 0xdf, 0x58, 0xe4, 0x49, 0x14, 0x15, 0x03, 0xed, 0xfa, 0xd4,
0x40, 0xa7, 0xf6, 0xca, 0xfb, 0x00, 0x4d, 0x5e, 0xe4, 0x55, 0x1d, 0x30,
0x45, 0xe2, 0xfc, 0x01, 0x48, 0x81, 0xe9, 0xf1, 0x1e, 0xfc, 0x21, 0x32,
0xed, 0x4b, 0xed, 0xfa, 0x2f, 0xd2, 0xfa, 0xfb, 0x4d, 0xa7, 0xed, 0xc7,
0x92, 0xdf, 0xe6, 0xdb, 0xf8, 0x1f, 0xd9, 0xfa, 0x91, 0xf5, 0xe5, 0xc5,
0x8c, 0x17, 0x0f, 0xb9, 0xd2, 0xc7, 0xfe, 0x68, 0xd3, 0x51, 0x2e, 0x49,
0x1f, 0xbd, 0x01, 0xeb, 0x31, 0x17, 0xf0, 0xef, 0xff, 0xb8, 0x5d, 0x62,
0x02, 0x0f, 0x1f, 0x78, 0x6a, 0xb0, 0xf9, 0xfe, 0x4f, 0xcc, 0xd3, 0xff,
0x0a, 0x96, 0x1e, 0x2c, 0xed, 0xbc, 0xf4, 0x0b, 0x42, 0xc8, 0xf1, 0xea,
0x6e, 0x58, 0xec, 0xc4, 0x99, 0xae, 0xdc, 0xd7, 0x12, 0x87, 0xd8, 0x06,
0xa2, 0xc2, 0xe6, 0xa2, 0x81, 0x24, 0xe9, 0xac, 0xce, 0xb6, 0x15, 0x6b,
0xba, 0x00, 0x19, 0x58, 0x29, 0xb6, 0xfe, 0x01, 0x25, 0x96, 0xd2, 0xec,
0x0e, 0x9c, 0x60, 0x5f, 0xe9, 0xf4, 0xf5, 0x69, 0x6b, 0xb5, 0xe1, 0xf6,

0x5e, 0xb7, 0xb1, 0xe5, 0x11, 0x9b, 0x18, 0x10, 0xe3, 0xe1, 0xe0, 0x0d,
0x4f, 0xa5, 0xde, 0xe5, 0x6f, 0xe2, 0xfb, 0x99, 0x82, 0xa5, 0xc9, 0xb6,
0x1f, 0x46, 0xf3, 0x04, 0xc6, 0xca, 0xd6, 0x97, 0x90, 0x1d, 0xc0, 0x95,
0xf0, 0x19, 0x30, 0x77, 0xc2, 0x3c, 0xfa, 0x24, 0x02, 0x4d, 0x06, 0x07,
0x15, 0x02, 0xb0, 0xe7, 0x27, 0x22, 0x67, 0x4d, 0xf1, 0xc2, 0xf4, 0x64,
0x38, 0x40, 0xdf, 0xf6, 0x3a, 0x43, 0xb8, 0xe1, 0x0d, 0x15, 0x11, 0xfe,
0xf5, 0xec, 0xf9, 0xe5, 0x22, 0x36, 0xe4, 0xfd, 0x6d, 0xbf, 0x0d, 0x8e,
0xb7, 0x15, 0xbf, 0x9f, 0x16, 0xad, 0x0a, 0x02, 0x8e, 0x14, 0xda, 0x9b,
0x8e, 0xc3, 0xa6, 0xca, 0xf5, 0x7f, 0x51, 0x56, 0xc1, 0xb3, 0xd9, 0x35,
0xf8, 0x7f, 0x04, 0x0a, 0x03, 0x3f, 0xbe, 0xee, 0x19, 0x68, 0x78, 0x50,
0xf9, 0xa7, 0xf7, 0x7f, 0x1d, 0x76, 0xdb, 0xe8, 0x33, 0xb9, 0xd7, 0xe7,
0xe8, 0x69, 0x15, 0xf7, 0xf5, 0xb2, 0xfe, 0xe8, 0xf3, 0x5b, 0xe2, 0x06,
0x6e, 0x09, 0x36, 0xb7, 0xcc, 0x38, 0xbf, 0x8a, 0x28, 0x14, 0x2e, 0x18,
0xa7, 0x26, 0xcb, 0xb2, 0x95, 0x37, 0xac, 0xcd, 0xd7, 0x51, 0x67, 0x44,
0xcd, 0x31, 0xde, 0x04, 0xe9, 0x6a, 0x00, 0x13, 0x0a, 0x0c, 0xdd, 0x16,
0xe0, 0x24, 0x7e, 0x49, 0xf1, 0xb5, 0x04, 0x52, 0x01, 0x50, 0xdd, 0xf5,
0x26, 0xc9, 0xf4, 0xf8, 0xd6, 0x31, 0x1b, 0xd0, 0xef, 0x03, 0x0a, 0xc0,
0xd4, 0x4f, 0xe2, 0xfd, 0x72, 0xf4, 0x5a, 0xc9, 0xd7, 0x31, 0xc0, 0x8e,
0x17, 0x5e, 0x57, 0x00, 0xb4, 0x3a, 0xc8, 0xd2, 0x92, 0x32, 0xcb, 0xd8,
0xc3, 0xa6, 0x63, 0x26, 0xcf, 0xbc, 0xe8, 0x57, 0x9b, 0xe9, 0xf7, 0x1c,
0xea, 0x12, 0xf1, 0xf7, 0xdb, 0xb9, 0x7f, 0x16, 0xf6, 0xe0, 0x08, 0x70,
0xa2, 0xed, 0xcc, 0xf1, 0x1e, 0x10, 0x04, 0xf7, 0xa9, 0xb7, 0x34, 0xaa,
0x0a, 0xdb, 0x2a, 0xa6, 0xb6, 0x10, 0xea, 0xf8, 0x5e, 0x06, 0x72, 0xdd,
0xd0, 0xb9, 0xd6, 0xa0, 0x10, 0x9f, 0x5a, 0x17, 0xb1, 0xe7, 0xc0, 0x01,
0x9d, 0x01, 0xe0, 0xe0, 0xaf, 0x9c, 0x46, 0xd8, 0xaf, 0xe8, 0xce, 0x02,
0x8a, 0xbb, 0xe4, 0xf6, 0xf3, 0x36, 0x07, 0xca, 0xcb, 0x87, 0x6e, 0xcc,

0xd6, 0x9e, 0x0a, 0x2a, 0x81, 0xd7, 0xcf, 0xc0, 0x04, 0xeb, 0x24, 0xcc,
0xc9, 0x95, 0x33, 0x81, 0xf7, 0xad, 0x1c, 0x9c, 0xa4, 0xd6, 0xf9, 0xe6,
0x3d, 0x84, 0x7f, 0xcc, 0xd4, 0xb0, 0xf4, 0xa2, 0xe9, 0x3c, 0x36, 0xee,
0xd5, 0xcf, 0xcd, 0x2d, 0x28, 0xbd, 0xff, 0xff, 0xc2, 0xbf, 0xff, 0xff,
0x04, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff,
0x31, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x48, 0xbd, 0xff, 0xff, 0x4c, 0xbd, 0xff, 0xff, 0xe6, 0xbf, 0xff, 0xff,
0x04, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x00, 0x8a, 0xfe, 0xff, 0xff,
0xa9, 0x00, 0x00, 0x00, 0xd0, 0xff, 0xff, 0xff, 0xd0, 0x00, 0x00, 0x00,
0x52, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00, 0x4f, 0xfb, 0xff, 0xff,
0x4a, 0xfd, 0xff, 0xff, 0x12, 0xc0, 0xff, 0xff, 0x04, 0x00, 0x00, 0x00,
0x80, 0x3e, 0x00, 0x00, 0xff, 0xf9, 0xfd, 0x0a, 0x07, 0x08, 0x07, 0x03,
0x07, 0xf2, 0xd1, 0x09, 0xf0, 0xe9, 0x28, 0x09, 0xdf, 0x05, 0xfa, 0xf0,
0xe8, 0xe3, 0x13, 0x0e, 0x08, 0xef, 0xd3, 0xee, 0x0f, 0xe8, 0xeb, 0x14,
0xf7, 0xed, 0xfd, 0x1f, 0xe8, 0xd5, 0xeb, 0xfc, 0x0e, 0xf4, 0xf7, 0x07,
0x05, 0xea, 0xf6, 0x1f, 0xf8, 0xdb, 0xdc, 0x0b, 0x03, 0xdd, 0xd8, 0xf3,
0x0f, 0x19, 0xe1, 0x09, 0xfc, 0xe4, 0x02, 0x04, 0xf1, 0x04, 0xeb, 0xf3,
0x1e, 0x06, 0xfd, 0x11, 0xfc, 0xfa, 0xf6, 0x1f, 0x0f, 0x02, 0xf5, 0xf7,
0xff, 0x24, 0xdf, 0xf7, 0xf8, 0xf3, 0xf6, 0xe9, 0xef, 0x03, 0xdd, 0xf2,
0x28, 0xe1, 0xf2, 0x22, 0xf4, 0x09, 0xf7, 0xf9, 0xf0, 0xd4, 0xf9, 0xee,
0xff, 0x14, 0xda, 0xf3, 0x11, 0xe2, 0xf6, 0x0c, 0xf2, 0xeb, 0xf8, 0xe8,
0xe3, 0x08, 0x02, 0x17, 0xf4, 0x0b, 0x0c, 0x27, 0xe6, 0x02, 0x03, 0xf9,
0x14, 0x18, 0xf6, 0xeb, 0x1f, 0x0c, 0xf1, 0xee, 0xfc, 0x08, 0xf0, 0xfe,
0xfd, 0xee, 0x17, 0xfd, 0x1c, 0xef, 0xfd, 0xde, 0x04, 0x05, 0xf0, 0x31,
0xfa, 0x0b, 0xdc, 0x0d, 0xed, 0xf5, 0xfa, 0xf4, 0x08, 0x0c, 0xd7, 0x1e,
0x15, 0x03, 0xf5, 0x02, 0xf4, 0xfb, 0xed, 0x01, 0xfe, 0xd6, 0x1f, 0xfd,

0xfd, 0x0e, 0xfa, 0x06, 0xf1, 0xf9, 0xe2, 0x16, 0xe9, 0xf1, 0x03, 0x0d,
0x0d, 0xdf, 0xf9, 0x1a, 0x0e, 0xf6, 0xfc, 0x0a, 0x19, 0xe2, 0xe0, 0x09,
0x15, 0xf0, 0xf1, 0x06, 0xf1, 0xe1, 0xef, 0x1a, 0x08, 0xe8, 0xfd, 0x12,
0x14, 0x06, 0xf1, 0xfc, 0xea, 0xfb, 0xf7, 0xea, 0x1d, 0x09, 0xfa, 0xf6,
0x08, 0xf2, 0xe7, 0xf8, 0xfc, 0x16, 0xf5, 0x0e, 0x08, 0xf9, 0x0a, 0x03,
0x26, 0xd8, 0x02, 0xf5, 0xf6, 0xf6, 0xef, 0x1f, 0xe4, 0xe2, 0xfb, 0x02,
0x1b, 0xe6, 0xde, 0x00, 0xf2, 0xed, 0xfb, 0x18, 0xe4, 0x16, 0x1a, 0x1d,
0xf1, 0xf6, 0xea, 0x16, 0x05, 0xde, 0xfb, 0x18, 0xf5, 0xe4, 0xfe, 0xe2,
0x1b, 0x1c, 0x0c, 0xe8, 0x02, 0xee, 0xfb, 0x07, 0x24, 0xf2, 0xe9, 0xfa,
0x0d, 0x05, 0xf1, 0x03, 0xfe, 0xf6, 0x19, 0x06, 0xff, 0xf9, 0x04, 0xfb,
0x15, 0xef, 0xf1, 0xf8, 0xe9, 0xe1, 0x10, 0x04, 0xfc, 0xe6, 0x1f, 0xed,
0x0b, 0xef, 0x00, 0x1e, 0xe6, 0x16, 0xf3, 0x09, 0xfd, 0x08, 0x08, 0x06,
0x06, 0x23, 0xdf, 0xfc, 0x08, 0xf4, 0xea, 0x0c, 0xf2, 0xe6, 0x18, 0xf5,
0x02, 0xf9, 0x50, 0x09, 0x01, 0xda, 0x0b, 0x05, 0x12, 0x18, 0xef, 0x04,
0x0e, 0xd9, 0xff, 0xdc, 0xf6, 0x16, 0xf9, 0xf4, 0xec, 0xff, 0xea, 0xe6,
0xfa, 0x0a, 0xed, 0xef, 0x02, 0xf0, 0x25, 0x21, 0xf1, 0x26, 0xf5, 0xed,
0x09, 0xea, 0xea, 0x24, 0xfa, 0x11, 0xfc, 0xdf, 0xf3, 0x0a, 0x28, 0x0c,
0x19, 0xff, 0xf5, 0xd6, 0x0e, 0xe2, 0x2a, 0x06, 0xfa, 0x03, 0xf9, 0xe6,
0xef, 0x23, 0xf9, 0xfa, 0xe6, 0xfe, 0xfc, 0x03, 0x06, 0x1a, 0xf9, 0x08,
0xe0, 0xe5, 0xff, 0x05, 0x01, 0xe7, 0x12, 0x02, 0x1d, 0x05, 0x03, 0x05,
0x0b, 0xee, 0xed, 0xfc, 0x0f, 0xf3, 0x02, 0xe0, 0x15, 0xdf, 0x02, 0xed,
0x10, 0x26, 0xef, 0x0d, 0x06, 0xee, 0xef, 0xf6, 0xeb, 0x11, 0x09, 0xf4,
0xf7, 0x06, 0x0f, 0x01, 0x2a, 0x0b, 0x01, 0xdd, 0xfc, 0xf4, 0xf1, 0x17,
0x03, 0x04, 0x07, 0xfc, 0x22, 0xfc, 0xde, 0xfe, 0x0b, 0x03, 0xf3, 0xfb,
0x0c, 0x25, 0x04, 0x19, 0x04, 0x03, 0x01, 0xfa, 0xfb, 0xf7, 0xf6, 0x0e,
0x15, 0x0e, 0x09, 0xff, 0x06, 0xfa, 0xfb, 0x1e, 0xfb, 0x05, 0x22, 0xf9,

0xfe, 0xf7, 0x1d, 0xed, 0xdf, 0x18, 0x09, 0xeb, 0xef, 0x04, 0x12, 0xea,
0xdf, 0xfb, 0xda, 0xf6, 0xdf, 0x17, 0xef, 0xef, 0xe1, 0x1a, 0xd9, 0xe2,
0xe2, 0xfc, 0x05, 0x11, 0xf6, 0xee, 0xe8, 0xf2, 0xe1, 0x08, 0x26, 0x04,
0xed, 0x03, 0xe0, 0xfb, 0xee, 0x0c, 0xee, 0xf6, 0x04, 0x2d, 0xf2, 0xd3,
0xf4, 0xe0, 0xf8, 0x0c, 0xfe, 0x11, 0x0b, 0xd7, 0xfd, 0x18, 0x07, 0x0d,
0x07, 0x08, 0xf4, 0xc6, 0x0a, 0x0a, 0x1f, 0x0c, 0xf4, 0x1d, 0x02, 0x0b,
0x09, 0x0e, 0x21, 0xff, 0x17, 0x0b, 0x0d, 0xf2, 0xed, 0xd7, 0x0a, 0xf8,
0x03, 0x06, 0xfa, 0xe5, 0xfd, 0x03, 0x14, 0x0f, 0xe9, 0x1a, 0xf4, 0xda,
0x01, 0xe6, 0x09, 0x06, 0x11, 0x0d, 0xfd, 0xeb, 0x16, 0x23, 0xfa, 0x00,
0x0b, 0x17, 0xf7, 0xda, 0xd7, 0x1b, 0xfa, 0x01, 0x03, 0x05, 0xfe, 0xd6,
0x02, 0xee, 0xee, 0x02, 0xf3, 0x06, 0xed, 0x03, 0xec, 0x01, 0xf2, 0x0f,
0x05, 0x17, 0x0b, 0xfb, 0x0f, 0x05, 0x03, 0x13, 0xff, 0x06, 0x02, 0xf5,
0xf4, 0x18, 0x2b, 0xf0, 0x00, 0x17, 0xfc, 0xfd, 0x05, 0x0b, 0x0e, 0x14,
0xe1, 0x24, 0x08, 0x24, 0xe6, 0xeb, 0x21, 0x12, 0xfb, 0x12, 0xe7, 0xf4,
0xe8, 0x0e, 0x18, 0xee, 0xf5, 0xf3, 0xd9, 0xf3, 0xdb, 0xec, 0x0c, 0x1e,
0xcf, 0x14, 0xdb, 0xe3, 0xdc, 0x02, 0x0c, 0xfb, 0xdb, 0x1b, 0xd0, 0xfe,
0xf9, 0xfe, 0x2a, 0xf5, 0x00, 0x0b, 0xcd, 0xe0, 0xe2, 0x0e, 0x04, 0xf8,
0xda, 0x1c, 0xe5, 0x0f, 0xe8, 0xf4, 0xf7, 0x15, 0x06, 0xf8, 0x02, 0xf7,
0x0f, 0xfb, 0x17, 0xf9, 0xda, 0x01, 0xda, 0xd1, 0xf6, 0x02, 0xfd, 0x16,
0xf1, 0xe4, 0xfa, 0x07, 0xee, 0x0a, 0xf3, 0xfd, 0xf2, 0x23, 0xf0, 0xe1,
0x0a, 0x1a, 0x12, 0x1f, 0xef, 0x27, 0x09, 0xf1, 0x0c, 0x13, 0x23, 0xfd,
0xf5, 0x03, 0xfe, 0x09, 0xfd, 0x16, 0xf8, 0x07, 0x08, 0x25, 0x08, 0xf8,
0xf6, 0x0a, 0xf1, 0xf5, 0x07, 0x09, 0x05, 0xcc, 0xf8, 0x08, 0x13, 0xf9,
0x1d, 0x11, 0x0f, 0xdc, 0xee, 0xf3, 0x27, 0xf9, 0xf9, 0x22, 0xfa, 0x0d,
0xe2, 0x13, 0xfb, 0x11, 0x03, 0x1e, 0xff, 0xfb, 0xed, 0xf1, 0x0e, 0x0b,
0x0f, 0x00, 0x06, 0xe0, 0x15, 0xf3, 0x13, 0xfc, 0x18, 0xf9, 0xff, 0x09,

0xfa, 0x1f, 0x12, 0xe5, 0xe2, 0x06, 0xf9, 0xf4, 0x07, 0x15, 0x0b, 0x04,
0xdb, 0x0d, 0xeb, 0xf3, 0xe6, 0x06, 0xe5, 0xee, 0xd8, 0x22, 0xd8, 0x10,
0xea, 0xf9, 0x1c, 0xf7, 0xd3, 0x11, 0xc3, 0xf8, 0xde, 0x05, 0x00, 0xe6,
0x07, 0xfd, 0xd3, 0x03, 0xea, 0xe0, 0x13, 0x14, 0xcf, 0xeb, 0xcd, 0xd3,
0xde, 0xf5, 0xf0, 0x0c, 0x0c, 0xfa, 0xeb, 0xd3, 0xfb, 0xfd, 0x08, 0xf9,
0xf4, 0x10, 0xfa, 0xd3, 0xf4, 0x11, 0x11, 0xf8, 0xef, 0xf8, 0xf8, 0xf1,
0xfc, 0xe1, 0xf7, 0x12, 0x04, 0xf4, 0xfb, 0xed, 0xef, 0x0c, 0xfd, 0x1c,
0xfe, 0x0e, 0xfd, 0xe2, 0xfe, 0x0a, 0x02, 0xfe, 0xe6, 0x1f, 0xef, 0xe5,
0xe6, 0xf8, 0x16, 0x27, 0xe8, 0x20, 0x05, 0xe3, 0xf1, 0xef, 0xee, 0xed,
0x0d, 0x11, 0x16, 0xfb, 0xf3, 0xff, 0x14, 0x01, 0xff, 0x15, 0x10, 0x02,
0xe5, 0x28, 0x29, 0x13, 0x13, 0x16, 0xe6, 0x00, 0xd2, 0x26, 0xfd, 0x03,
0x04, 0x05, 0x07, 0x06, 0xf1, 0x0e, 0x05, 0x0d, 0xe2, 0x0f, 0x02, 0xe1,
0x07, 0xf7, 0x1c, 0xfa, 0x14, 0x30, 0xf7, 0xee, 0x00, 0xfa, 0x3d, 0x06,
0x1c, 0x04, 0x06, 0x07, 0x05, 0x1a, 0x10, 0xf6, 0xee, 0x0a, 0xeb, 0x04,
0xeb, 0xdf, 0x1d, 0x09, 0xd5, 0xe8, 0xd6, 0xf4, 0xf0, 0x0f, 0x1d, 0xea,
0xf2, 0xf8, 0xa6, 0x0b, 0xdc, 0x09, 0x08, 0x24, 0xee, 0x24, 0xaa, 0xe4,
0xcb, 0x15, 0xef, 0xe7, 0xe9, 0x0c, 0xcf, 0x06, 0xe3, 0x12, 0x11, 0x00,
0x07, 0x14, 0xd7, 0xde, 0xf6, 0x0f, 0x0b, 0x04, 0xfb, 0x0d, 0xf8, 0x0d,
0xf6, 0x1b, 0xf1, 0x21, 0xdd, 0xfc, 0xf4, 0xe9, 0xf8, 0xe8, 0xf7, 0x06,
0x03, 0x1e, 0xce, 0xe1, 0xea, 0xf6, 0x05, 0xf9, 0x16, 0x15, 0x04, 0xe0,
0x14, 0xf7, 0x1e, 0x1c, 0x0a, 0x27, 0xef, 0xf3, 0x0f, 0xf3, 0xee, 0x04,
0xf8, 0xf1, 0x07, 0xe3, 0x05, 0x0b, 0x00, 0x1c, 0x15, 0x27, 0x07, 0xf7,
0xfa, 0x0b, 0xfa, 0xfa, 0x17, 0x13, 0xe1, 0xf5, 0xfb, 0x0c, 0x21, 0x2f,
0xd7, 0xfb, 0xf5, 0xfd, 0xd3, 0xf4, 0x07, 0x0e, 0xfd, 0x0b, 0xfc, 0xfa,
0xf5, 0x0e, 0x02, 0xfa, 0xfa, 0x19, 0xfd, 0xfa, 0xfc, 0x13, 0x24, 0x0c,
0xe4, 0x31, 0xf8, 0x12, 0xf4, 0x04, 0x18, 0x29, 0x27, 0x19, 0xfc, 0x08,

0x11, 0xe3, 0x07, 0xfe, 0x26, 0x40, 0x05, 0x02, 0x04, 0x02, 0x0f, 0xee,
0xf4, 0x27, 0xea, 0xf4, 0xf5, 0x11, 0x26, 0x0b, 0xe7, 0x05, 0xd2, 0xf6,
0xea, 0xfa, 0x0b, 0xf9, 0xfa, 0x16, 0xba, 0x00, 0xfb, 0x0d, 0x0b, 0xf9,
0xe6, 0xf6, 0xc5, 0xf8, 0xf6, 0x01, 0x0f, 0xed, 0xed, 0x13, 0xcd, 0x0d,
0xda, 0x06, 0x17, 0xee, 0x07, 0x1d, 0xb8, 0xfa, 0xe2, 0xea, 0xf2, 0xee,
0x04, 0x00, 0xdc, 0xd0, 0xfb, 0xf5, 0xec, 0xfe, 0xf1, 0x0d, 0xf0, 0xdb,
0xf9, 0x0d, 0x03, 0x03, 0x0e, 0x0a, 0xda, 0xd6, 0x01, 0xf2, 0x06, 0x14,
0x1c, 0x1f, 0xe8, 0xe8, 0x0e, 0xfd, 0x0c, 0xf5, 0xf3, 0x3d, 0xf3, 0x05,
0x10, 0xfa, 0x1b, 0x18, 0x08, 0x36, 0x09, 0xf1, 0xeb, 0xf9, 0x22, 0x01,
0xf3, 0xf7, 0xff, 0xf0, 0x0c, 0xe9, 0x01, 0x29, 0x21, 0x15, 0x03, 0xee,
0xe9, 0x1a, 0xf7, 0x15, 0x06, 0x25, 0xfa, 0xf0, 0xe4, 0xf1, 0x1f, 0x01,
0xdc, 0x2d, 0xce, 0xe9, 0xea, 0x0b, 0x06, 0x2c, 0x0a, 0x30, 0xe7, 0x09,
0xf4, 0xf0, 0x10, 0x29, 0xf9, 0x3d, 0xe7, 0xdc, 0xe4, 0xf7, 0x3b, 0x27,
0x23, 0x3a, 0x0a, 0x06, 0x0e, 0xfd, 0x2c, 0x07, 0x2b, 0x1c, 0xfa, 0x00,
0xf9, 0x11, 0xea, 0x14, 0xeb, 0xfc, 0x18, 0x03, 0xf1, 0x16, 0x12, 0x04,
0xcf, 0x12, 0xdd, 0xe4, 0x0e, 0xf0, 0x09, 0xe8, 0xf3, 0xfb, 0xa8, 0xf9,
0xee, 0xfb, 0x1e, 0x1d, 0xfd, 0x05, 0xab, 0xe5, 0xff, 0x01, 0xfe, 0x04,
0xf9, 0x02, 0xb9, 0xdc, 0xdf, 0x05, 0xf1, 0xef, 0xf1, 0x1e, 0xc7, 0xee,
0xf7, 0x1e, 0x00, 0x00, 0xf8, 0x10, 0xec, 0xe8, 0x04, 0x0f, 0xf6, 0xff,
0x04, 0x09, 0xe0, 0x0a, 0x0e, 0xe4, 0xf0, 0xf1, 0x16, 0x2b, 0xd3, 0xe1,
0x0a, 0xef, 0xf9, 0xfe, 0x0b, 0x22, 0xf5, 0x01, 0x0a, 0xf8, 0x02, 0x00,
0x17, 0x19, 0xf3, 0x05, 0x21, 0xfa, 0xee, 0xee, 0x12, 0xf2, 0xfa, 0xf5,
0x05, 0x12, 0xee, 0xe4, 0x28, 0xfa, 0xf1, 0x03, 0x15, 0x16, 0x18, 0xfd,
0x0f, 0x21, 0x04, 0xf4, 0xe5, 0x0c, 0x06, 0x13, 0xde, 0x36, 0xe8, 0xfb,
0xe7, 0xfd, 0xf6, 0x12, 0x0e, 0x1d, 0xea, 0xf8, 0xd4, 0xe8, 0x19, 0x07,
0xe5, 0x1c, 0xf7, 0x0c, 0xef, 0x05, 0x0f, 0x09, 0xdd, 0x1a, 0xea, 0xd7,

0xf9, 0xf9, 0x12, 0x17, 0x2e, 0x10, 0x08, 0xfe, 0x14, 0xf5, 0x1d, 0xfa,
0x06, 0x33, 0xed, 0xfe, 0xf7, 0x11, 0xf0, 0x15, 0xe2, 0x24, 0xf6, 0x0a,
0xe2, 0xfc, 0x23, 0x12, 0xdd, 0x11, 0xfd, 0xe5, 0x08, 0xff, 0x15, 0xf6,
0xf1, 0x1b, 0xae, 0xfe, 0xe6, 0x15, 0x2c, 0x2d, 0x15, 0x15, 0xc5, 0xf8,
0xea, 0xe7, 0x07, 0x04, 0xfe, 0x28, 0xa1, 0xf2, 0xe1, 0xf9, 0xf8, 0xff,
0xf4, 0x22, 0xb4, 0xdb, 0x03, 0x20, 0xe6, 0xf3, 0x0e, 0x19, 0xe3, 0x0a,
0xfa, 0xee, 0xf3, 0xe5, 0xd8, 0xf9, 0xf1, 0xde, 0x06, 0x05, 0xf2, 0xf5,
0xe7, 0x16, 0xd8, 0xfe, 0x07, 0xea, 0xee, 0x0e, 0xfa, 0xff, 0xdb, 0xe7,
0x03, 0xed, 0x01, 0xfd, 0x09, 0x1a, 0xfa, 0xe6, 0x05, 0x10, 0xe9, 0x01,
0x1f, 0x13, 0xf7, 0xf6, 0xfb, 0x13, 0xff, 0xdb, 0xed, 0xfe, 0x0a, 0x10,
0x09, 0x29, 0xf5, 0x04, 0xf5, 0x26, 0x0d, 0x0c, 0xf9, 0x16, 0xfa, 0x02,
0xf4, 0x2e, 0xde, 0xf5, 0xe1, 0x1d, 0xfb, 0x02, 0x0b, 0x23, 0x07, 0xea,
0xd9, 0x0a, 0xf3, 0x0a, 0x0f, 0x1e, 0xe7, 0xf1, 0xd7, 0x0b, 0xf6, 0xff,
0x0d, 0x24, 0xcc, 0x0a, 0xee, 0xda, 0x14, 0x12, 0x11, 0x29, 0xf4, 0x1a,
0xef, 0x0b, 0xfa, 0xec, 0x0c, 0x1b, 0xf4, 0xff, 0xf5, 0xef, 0x0f, 0x10,
0xd4, 0x04, 0xf9, 0xf8, 0xec, 0xf9, 0x21, 0x05, 0xd3, 0x27, 0xf3, 0x17,
0xff, 0xf6, 0x15, 0xf9, 0xed, 0x0a, 0xac, 0x02, 0xfd, 0xfb, 0x04, 0x29,
0x06, 0x03, 0xb8, 0xe6, 0xd5, 0x17, 0x09, 0x1b, 0xf6, 0x1b, 0xab, 0xdc,
0xdf, 0xfd, 0x06, 0x09, 0x09, 0x37, 0xbb, 0xed, 0x19, 0xd7, 0xe2, 0xdd,
0x05, 0x01, 0xec, 0xfb, 0xe4, 0x0e, 0xeb, 0xf0, 0x03, 0x17, 0x04, 0xeb,
0x09, 0xee, 0xeb, 0xe7, 0x0c, 0x16, 0xcb, 0x0e, 0x17, 0xd8, 0xe1, 0xf8,
0x2b, 0x19, 0xde, 0xeb, 0x10, 0xf2, 0xff, 0xf8, 0xee, 0x0e, 0xe7, 0xf0,
0x15, 0x08, 0xf8, 0xdf, 0x06, 0x0d, 0xf9, 0x14, 0xfa, 0x0b, 0x04, 0xfd,
0x15, 0x23, 0x20, 0xff, 0xfd, 0x1d, 0x0c, 0xf1, 0xfe, 0x15, 0x0a, 0x02,
0xed, 0xfe, 0xfb, 0x04, 0xfb, 0x1e, 0xdd, 0x05, 0xe0, 0x16, 0xf9, 0xf6,
0xfd, 0x32, 0xdc, 0xf2, 0xd3, 0x08, 0xf4, 0xec, 0x17, 0x25, 0xe2, 0xf0,

0xee, 0xf1, 0x0d, 0xfe, 0x13, 0x2d, 0x01, 0x11, 0xd4, 0xe4, 0x07, 0xfb,
0x32, 0x11, 0x14, 0x07, 0xd7, 0x02, 0x10, 0xeb, 0x2b, 0x1d, 0x01, 0xfc,
0xf3, 0xf0, 0x13, 0x1a, 0xdb, 0x20, 0x00, 0xf0, 0xf0, 0x05, 0x16, 0x03,
0xd4, 0xe3, 0xc2, 0xf0, 0x06, 0x02, 0x1e, 0x0a, 0xec, 0x1f, 0xab, 0xea,
0xfa, 0xe3, 0x20, 0x22, 0x03, 0x1b, 0xb3, 0x0e, 0xe3, 0xf3, 0x1d, 0x27,
0xe3, 0x10, 0xa7, 0xda, 0xf3, 0x00, 0x0a, 0x0a, 0x04, 0xfb, 0xb2, 0x0f,
0x0c, 0xf5, 0x07, 0xff, 0x13, 0x1e, 0xdb, 0xf6, 0xf9, 0xef, 0xe8, 0xe7,
0xfb, 0x18, 0xeb, 0xec, 0x09, 0xda, 0xf1, 0xf0, 0x0b, 0x04, 0xe1, 0xfa,
0x1c, 0x25, 0xee, 0x01, 0x0b, 0x29, 0xd7, 0x0c, 0x04, 0x0b, 0xef, 0xfd,
0x1c, 0xfc, 0xf1, 0xfb, 0x0b, 0x0f, 0xdf, 0xed, 0x17, 0x38, 0x0c, 0xd7,
0xff, 0xfd, 0x01, 0xfc, 0xfb, 0xfb, 0x18, 0x1a, 0x18, 0xe3, 0xf9, 0xf4,
0xfa, 0x20, 0x06, 0x09, 0x11, 0x08, 0x1d, 0xf8, 0xfa, 0x1d, 0xf5, 0x1c,
0xf5, 0xfe, 0x03, 0x07, 0xe4, 0x33, 0xc8, 0x0c, 0xe1, 0x13, 0xff, 0xe5,
0x10, 0x2c, 0xd3, 0xf0, 0xed, 0x04, 0x07, 0x01, 0xf1, 0x16, 0xe0, 0x13,
0xfa, 0x11, 0x07, 0xfa, 0x19, 0x16, 0x01, 0x00, 0x07, 0x26, 0x00, 0xec,
0x1d, 0x23, 0x05, 0xf4, 0x07, 0x17, 0x2c, 0x1d, 0xee, 0xf0, 0x0c, 0x09,
0xe3, 0x1a, 0x24, 0x0b, 0xf3, 0x1e, 0xce, 0xfe, 0xfe, 0x12, 0x21, 0x1a,
0xf6, 0x23, 0xc3, 0x03, 0xf4, 0x10, 0x1a, 0x2a, 0xf4, 0x08, 0xbf, 0xff,
0x04, 0xf4, 0x0b, 0x1d, 0x1a, 0xf8, 0xcc, 0x00, 0xf7, 0x13, 0xf4, 0xfd,
0xf4, 0x19, 0xbd, 0xef, 0x0c, 0x0d, 0x02, 0xfc, 0x12, 0x13, 0xe9, 0xe7,
0xf5, 0xfa, 0xfa, 0xf6, 0x1a, 0x2e, 0xce, 0xd4, 0x01, 0x12, 0xfd, 0xfc,
0x26, 0x10, 0xcc, 0xe7, 0xee, 0x13, 0xee, 0xff, 0xef, 0xea, 0x00, 0x0e,
0x1a, 0x17, 0x04, 0x0c, 0x04, 0x0c, 0xe6, 0xf3, 0xf6, 0xdb, 0xdd, 0x04,
0xf4, 0x22, 0x11, 0x16, 0xf3, 0x07, 0xec, 0xf8, 0xf2, 0x07, 0x03, 0x02,
0xf5, 0x0a, 0xf6, 0x02, 0x1d, 0x1b, 0x11, 0x06, 0xf8, 0x06, 0x02, 0xea,
0xf3, 0x1d, 0xce, 0x00, 0xed, 0xf9, 0xef, 0xf6, 0xec, 0x22, 0xc7, 0xf0,

0xed, 0xdb, 0xe0, 0x02, 0x11, 0x07, 0xe8, 0xf0, 0xd1, 0xed, 0xff, 0xfd,
0x0c, 0x2e, 0xd4, 0xed, 0xec, 0x0e, 0xf1, 0x07, 0x01, 0x0e, 0x0e, 0xfe,
0xda, 0x0b, 0x0a, 0x0a, 0x1f, 0x2e, 0x13, 0x07, 0x00, 0x07, 0x14, 0x21,
0xe9, 0xfc, 0xf0, 0x1e, 0xd7, 0xea, 0x34, 0x07, 0xc6, 0x0c, 0xd4, 0xec,
0xfd, 0x06, 0x24, 0x0a, 0xf3, 0x15, 0xaf, 0xff, 0xe9, 0xf1, 0x0d, 0x3e,
0xe9, 0x18, 0xba, 0x13, 0xed, 0xd7, 0x0b, 0x31, 0x05, 0x0e, 0xaf, 0x13,
0xd6, 0x0e, 0x10, 0x02, 0x02, 0x14, 0xcb, 0xd5, 0xf9, 0x0c, 0xf9, 0x0e,
0x1f, 0x24, 0xd5, 0xeb, 0xff, 0xf1, 0xf5, 0x0c, 0x08, 0x07, 0xf4, 0xd7,
0x06, 0x10, 0xe8, 0xef, 0xfc, 0x2f, 0xee, 0xf1, 0x18, 0xf8, 0xf4, 0x02,
0x11, 0x21, 0xd3, 0x12, 0x14, 0xe4, 0xf4, 0x02, 0x05, 0x24, 0xca, 0xf2,
0xf3, 0xeb, 0xe7, 0xf8, 0x16, 0x1a, 0xeb, 0x0d, 0x05, 0x16, 0xf1, 0xec,
0x11, 0x1c, 0x09, 0x1e, 0xe0, 0xe6, 0xfa, 0x0e, 0x0d, 0x2a, 0xea, 0x2e,
0xed, 0xf9, 0xf7, 0x16, 0x09, 0x05, 0xdd, 0xd6, 0x02, 0xeb, 0xf5, 0xf3,
0xe4, 0x3b, 0xed, 0x04, 0xe0, 0x0e, 0xfd, 0x09, 0xfd, 0x35, 0xdc, 0x18,
0xf3, 0x04, 0xfa, 0x05, 0x15, 0x34, 0xe5, 0xe1, 0xe4, 0xf4, 0xe0, 0xf9,
0x08, 0x32, 0x04, 0x08, 0xf4, 0x0f, 0xff, 0x08, 0x09, 0x2f, 0x06, 0x02,
0xfd, 0x05, 0x0c, 0x24, 0xe3, 0x1e, 0xf5, 0x0c, 0xdd, 0xf8, 0x18, 0x20,
0xd8, 0x14, 0xef, 0xf4, 0x17, 0x08, 0x25, 0x14, 0x04, 0x06, 0xb0, 0xf5,
0xf5, 0x09, 0x0f, 0x3e, 0xff, 0x28, 0xb3, 0xf5, 0x19, 0xd8, 0x14, 0x21,
0xd9, 0xf7, 0xb7, 0xe5, 0xfe, 0xe7, 0x07, 0x1e, 0x04, 0x15, 0xc5, 0xf9,
0x14, 0x20, 0xeb, 0x01, 0x01, 0x18, 0xce, 0x00, 0xe6, 0xe2, 0xf7, 0xfb,
0xf3, 0x0d, 0xd3, 0xf3, 0x04, 0xf8, 0xf0, 0x03, 0xf1, 0x25, 0xb5, 0xef,
0x05, 0xe0, 0x01, 0xf6, 0x04, 0x16, 0xd1, 0x01, 0x0a, 0x21, 0x01, 0x05,
0x0e, 0x01, 0xf0, 0x0a, 0xf3, 0x00, 0x03, 0xf8, 0xfa, 0x03, 0x0b, 0xde,
0xfe, 0xff, 0xfb, 0xea, 0x09, 0x02, 0xf5, 0xe8, 0xe7, 0x08, 0x00, 0xf5,
0xf8, 0x0f, 0x13, 0xfa, 0xeb, 0xe8, 0xfb, 0x1f, 0x08, 0x16, 0xe6, 0xfa,

0xe1, 0x00, 0x03, 0xdd, 0xf1, 0x26, 0xe5, 0x1d, 0xd9, 0xff, 0xf2, 0xf8,
0xff, 0x33, 0xea, 0xe5, 0x03, 0x0c, 0x07, 0xf9, 0xf8, 0x0f, 0xe1, 0x1e,
0xdd, 0x0f, 0x00, 0xf1, 0x06, 0x21, 0x09, 0x05, 0xf3, 0xec, 0xe6, 0x04,
0x07, 0x32, 0xf1, 0xf9, 0xf2, 0x01, 0x18, 0x1f, 0xd2, 0xe2, 0x0a, 0xf4,
0xca, 0xfc, 0x28, 0x16, 0xc2, 0x10, 0xf2, 0xfc, 0x08, 0xe9, 0x2a, 0x0f,
0xfa, 0xf5, 0xa9, 0x07, 0xec, 0xe9, 0x19, 0x43, 0x0b, 0x1c, 0xa6, 0xe9,
0xf4, 0x16, 0x0d, 0x2b, 0xfc, 0x11, 0x9a, 0xe1, 0xf1, 0x1c, 0xf5, 0x0f,
0xe4, 0x18, 0xc0, 0xd9, 0x14, 0x26, 0xe6, 0xf8, 0x0a, 0x17, 0xec, 0xfb,
0xe1, 0x22, 0xdf, 0xf2, 0xfe, 0x1e, 0xd4, 0xeb, 0xd7, 0x0e, 0x08, 0xf6,
0xef, 0xfc, 0xe6, 0xd4, 0xf7, 0x0b, 0xfb, 0xf5, 0x01, 0x25, 0xd7, 0xfb,
0x0d, 0xfe, 0xff, 0xf3, 0x1d, 0x32, 0xfe, 0xee, 0x12, 0xf2, 0x0c, 0xec,
0x02, 0x10, 0xef, 0x01, 0xf2, 0x0b, 0xf3, 0xf7, 0xfa, 0x25, 0xfb, 0x0d,
0x11, 0x15, 0x04, 0xfc, 0x0c, 0x21, 0x12, 0x29, 0x00, 0xfa, 0xf6, 0xf5,
0x06, 0x22, 0xea, 0xe2, 0xee, 0x00, 0xfd, 0xf0, 0x0b, 0x1d, 0xd3, 0xe4,
0xe4, 0x0a, 0xfc, 0xe8, 0xea, 0x2c, 0xed, 0xed, 0xef, 0xe8, 0xf2, 0x05,
0xfd, 0x15, 0xd8, 0xda, 0xca, 0xee, 0xfa, 0x00, 0xfe, 0x0e, 0xf2, 0xf0,
0x0e, 0xf5, 0x04, 0x03, 0x1d, 0x2b, 0xee, 0x05, 0x0f, 0x10, 0x13, 0x35,
0xe2, 0x04, 0x10, 0xdf, 0xcf, 0xeb, 0x40, 0x26, 0xe4, 0x03, 0xf3, 0xf9,
0xf5, 0x14, 0x24, 0x2a, 0xdf, 0xfe, 0xab, 0xe5, 0xfe, 0x1c, 0x27, 0x35,
0xdb, 0xff, 0xac, 0x01, 0xf6, 0xfc, 0x19, 0x1a, 0x11, 0x1f, 0xa8, 0xf5,
0x02, 0x0f, 0x1a, 0x1f, 0xf7, 0xf2, 0xa2, 0x00, 0x15, 0x22, 0xe4, 0x13,
0x00, 0x09, 0xd9, 0xd5, 0x02, 0x19, 0xfd, 0xf8, 0xe7, 0xff, 0xfb, 0xe0,
0xef, 0xf7, 0xee, 0xf3, 0xf3, 0x19, 0xb0, 0xdf, 0x00, 0x0f, 0x08, 0xf3,
0x15, 0x17, 0xec, 0x0f, 0x11, 0x14, 0x02, 0x08, 0x10, 0x17, 0xe6, 0x08,
0xf7, 0x00, 0xed, 0xf7, 0x29, 0x07, 0x10, 0x05, 0x05, 0xe7, 0xed, 0xf4,
0xf9, 0x15, 0xf9, 0xf0, 0x08, 0x00, 0x03, 0x09, 0x21, 0x28, 0xf6, 0x0e,

0xfb, 0xf3, 0x03, 0xf7, 0x0f, 0x0c, 0xf0, 0xf5, 0xe3, 0xd8, 0xf8, 0xf2,
0x09, 0x1c, 0xe7, 0xfb, 0xe4, 0xf6, 0xfa, 0xf8, 0xf1, 0x42, 0xf6, 0xda,
0xdd, 0xd7, 0xfa, 0xff, 0x2f, 0x2c, 0xda, 0x0a, 0xde, 0xec, 0xf1, 0x14,
0xfb, 0x1d, 0xeb, 0xee, 0xf2, 0xeb, 0xf3, 0xed, 0x0e, 0x35, 0xf0, 0x06,
0x19, 0x04, 0x2f, 0x23, 0xe2, 0x07, 0x13, 0x0f, 0xe9, 0xf0, 0x22, 0x2e,
0xd9, 0x1a, 0xcb, 0xed, 0xfd, 0x04, 0x27, 0x1e, 0xf6, 0x07, 0x96, 0xd6,
0xd8, 0x11, 0x18, 0x56, 0xd2, 0xfb, 0x92, 0xfc, 0x0b, 0x0a, 0x17, 0x2c,
0xe5, 0x04, 0xa2, 0xf8, 0xe2, 0x04, 0x1a, 0x0d, 0xeb, 0x11, 0xa2, 0xe5,
0xe5, 0xf8, 0x02, 0xf7, 0x17, 0x03, 0xca, 0xe9, 0x0c, 0x1f, 0xfe, 0xf5,
0x18, 0x12, 0xdd, 0x08, 0x15, 0xff, 0xfc, 0xf6, 0xe1, 0x1d, 0xe2, 0xe1,
0xfe, 0xfc, 0x03, 0xff, 0xf2, 0x23, 0xd2, 0x01, 0x13, 0xdd, 0xf3, 0xf4,
0xf2, 0x07, 0xef, 0x03, 0x15, 0x21, 0xd8, 0xf8, 0x09, 0xf3, 0xe8, 0xea,
0xe8, 0xf2, 0x08, 0xf0, 0x04, 0x1a, 0xf2, 0x19, 0xfb, 0x1b, 0x15, 0xfc,
0x1d, 0x30, 0xe5, 0x1e, 0x09, 0xe8, 0xe9, 0x09, 0xf7, 0x2a, 0xe1, 0x0e,
0x00, 0x21, 0xf3, 0xff, 0xfb, 0x01, 0xdf, 0xf2, 0xfe, 0xf4, 0xfc, 0xf0,
0x0b, 0x0b, 0xdd, 0xe4, 0xd2, 0x14, 0xf7, 0xfe, 0x0b, 0x39, 0x01, 0xe6,
0xe4, 0x27, 0xfa, 0xe4, 0x04, 0x2c, 0xe2, 0x04, 0xf5, 0x07, 0xf2, 0x03,
0xf0, 0x10, 0xf5, 0xf6, 0xfc, 0x16, 0x22, 0x1b, 0xf8, 0x11, 0xe4, 0x09,
0xf6, 0xf0, 0x41, 0x1e, 0xcf, 0x04, 0xea, 0xee, 0x0e, 0xf6, 0x1b, 0x2f,
0xc7, 0xf1, 0xba, 0xef, 0x0f, 0x16, 0x1e, 0x39, 0x05, 0x1e, 0x90, 0xe6,
0x0d, 0xfa, 0x22, 0x3f, 0xe3, 0x23, 0xa5, 0xe3, 0xe9, 0x0f, 0x05, 0x27,
0x02, 0x11, 0x99, 0x05, 0xfa, 0x05, 0x03, 0x01, 0xff, 0x26, 0xd3, 0xf7,
0xf7, 0xf9, 0x05, 0xf4, 0xef, 0x23, 0xd2, 0xdd, 0x05, 0x08, 0xfa, 0xff,
0x03, 0x04, 0xbd, 0xd7, 0x14, 0x06, 0xef, 0x06, 0xe5, 0x05, 0xea, 0xea,
0x02, 0xfd, 0x0d, 0x00, 0x08, 0xff, 0xe7, 0xfb, 0xfe, 0x13, 0xfe, 0xec,
0xf9, 0x02, 0xf3, 0xff, 0xff, 0x08, 0x04, 0xed, 0x19, 0x1d, 0xfa, 0x0a,

0x0d, 0xf2, 0x0f, 0xec, 0x25, 0x1c, 0xec, 0x0b, 0x01, 0xff, 0x01, 0xf6,
0x08, 0x09, 0xe8, 0xe2, 0xec, 0x23, 0xe5, 0xe9, 0xf0, 0x2e, 0xbd, 0xe1,
0xef, 0x14, 0xe9, 0xf6, 0xf5, 0x1d, 0xdc, 0xe3, 0xd7, 0xfc, 0xf9, 0xf2,
0xfe, 0x24, 0xf2, 0x05, 0xd5, 0xed, 0xe9, 0xf9, 0xfa, 0x2d, 0xf0, 0xfe,
0xee, 0xf2, 0xe8, 0xf7, 0x06, 0x14, 0x01, 0x10, 0x06, 0xf3, 0x0e, 0x0e,
0xc2, 0x1d, 0xf2, 0x1c, 0xed, 0xe3, 0x53, 0x21, 0xb8, 0x0c, 0xde, 0x03,
0x15, 0xeb, 0x46, 0x39, 0xdf, 0xf6, 0xa3, 0xee, 0xf6, 0xe0, 0x33, 0x50,
0xdd, 0x27, 0x9f, 0x07, 0x13, 0xe2, 0x1f, 0x35, 0xed, 0x1f, 0xb7, 0x07,
0x11, 0xed, 0x17, 0x28, 0xf4, 0x20, 0xc1, 0xec, 0xef, 0x16, 0x02, 0xfa,
0xe0, 0x1b, 0xf7, 0xdb, 0xfd, 0x0a, 0xe7, 0xfb, 0xe7, 0x25, 0xe2, 0xe7,
0xf8, 0xf0, 0xee, 0xe9, 0x02, 0x06, 0xc9, 0xe4, 0x14, 0xe3, 0xe2, 0xf7,
0xf8, 0xfd, 0xdd, 0xe2, 0x08, 0x0a, 0xe4, 0x05, 0xf5, 0x16, 0xe7, 0x01,
0x00, 0x1c, 0xe7, 0xf0, 0xf6, 0x19, 0xfe, 0x0c, 0xf2, 0x06, 0x03, 0xe8,
0x0b, 0xfe, 0xe3, 0x19, 0x08, 0x1a, 0x10, 0xfd, 0x00, 0x21, 0xf0, 0xeb,
0x18, 0x02, 0xf3, 0x04, 0xf0, 0x18, 0xdb, 0x05, 0x01, 0xde, 0xed, 0xe9,
0x23, 0x15, 0xaf, 0xe6, 0xf1, 0x0a, 0xe6, 0xea, 0x01, 0x18, 0xd8, 0xfd,
0xf1, 0xe6, 0xec, 0xf5, 0x0e, 0x1e, 0xcc, 0xfc, 0xe7, 0x00, 0xe9, 0x11,
0x00, 0x30, 0xf9, 0x14, 0xf4, 0x19, 0xdd, 0xf7, 0xf7, 0x2f, 0xf4, 0xf2,
0xff, 0x27, 0x15, 0x1c, 0xbc, 0x2f, 0xe9, 0x14, 0xf5, 0xe8, 0x44, 0x30,
0xe8, 0x1d, 0xe4, 0x18, 0x11, 0x00, 0x0c, 0x2b, 0xf3, 0x29, 0x96, 0xe0,
0x06, 0xee, 0x3e, 0x55, 0xdc, 0x13, 0x98, 0xdf, 0xf0, 0xfe, 0x17, 0x33,
0xe8, 0x09, 0xa3, 0x07, 0xef, 0x0e, 0x1d, 0x37, 0xdd, 0xfe, 0xb5, 0x00,
0xf7, 0xe0, 0xea, 0xfd, 0xfd, 0x19, 0xbc, 0xfd, 0x15, 0xfe, 0x01, 0xf3,
0xd5, 0x20, 0xbf, 0xe3, 0x15, 0x0e, 0xf0, 0xf6, 0xf2, 0x14, 0xcc, 0xf0,
0xf7, 0x04, 0xf2, 0xff, 0x0b, 0x02, 0xd2, 0xd8, 0xfa, 0xfc, 0xe5, 0x02,
0x00, 0xfb, 0xf0, 0xdc, 0x1e, 0x10, 0x02, 0x01, 0x00, 0x18, 0xe9, 0xdb,

0x1e, 0xf6, 0xfc, 0x03, 0xef, 0x0a, 0x00, 0x16, 0x00, 0x0f, 0xf4, 0x16,
0xfa, 0x0b, 0xe2, 0xfa, 0xe0, 0x07, 0xfb, 0x02, 0x21, 0x0e, 0xdd, 0x0b,
0xea, 0xf0, 0xeb, 0xfb, 0x19, 0x09, 0xd4, 0xf2, 0xef, 0x0b, 0x00, 0xeb,
0x1a, 0x2f, 0xea, 0x06, 0x03, 0xf6, 0xf8, 0xfb, 0xfe, 0x1d, 0xea, 0xdd,
0xed, 0xfd, 0xfb, 0xe7, 0xfe, 0x18, 0xf4, 0xfc, 0x0b, 0xf6, 0xfc, 0x0b,
0xfb, 0x28, 0x07, 0xff, 0x07, 0x1e, 0x03, 0x21, 0xcf, 0x22, 0x05, 0xe6,
0xea, 0xe7, 0x43, 0x2e, 0xe7, 0x14, 0xfb, 0x0a, 0x1e, 0xfe, 0x2c, 0x24,
0xd5, 0xfd, 0x9e, 0xd1, 0xf2, 0x1c, 0x32, 0x51, 0x01, 0xf3, 0xac, 0xe1,
0xf4, 0xe5, 0x1c, 0x37, 0xf1, 0x0f, 0xa7, 0xdb, 0x00, 0xf6, 0x0f, 0x18,
0xe1, 0x10, 0xc9, 0xc5, 0xe8, 0xeb, 0xf2, 0xfd, 0xf6, 0x02, 0xc2, 0xff,
0x00, 0x19, 0x03, 0x0f, 0x02, 0x22, 0xd4, 0xe7, 0x07, 0x0f, 0xe5, 0x1a,
0x09, 0x0b, 0xdc, 0xd2, 0x00, 0x05, 0xee, 0xf8, 0xdc, 0x14, 0xd0, 0x0a,
0x0a, 0xfa, 0xeb, 0x04, 0xf3, 0x06, 0xde, 0x05, 0xfb, 0xfd, 0xe3, 0xec,
0xfd, 0x14, 0xd7, 0x11, 0x0e, 0xe6, 0x06, 0xec, 0xde, 0x22, 0xd7, 0x00,
0x03, 0xf5, 0xf5, 0x0d, 0x01, 0x05, 0xea, 0x0b, 0x16, 0x04, 0xff, 0x13,
0xf3, 0x12, 0xd2, 0xdf, 0x0b, 0xe4, 0x06, 0xf6, 0x08, 0x2d, 0xd3, 0xd6,
0xe7, 0x0a, 0xec, 0xff, 0xfe, 0x01, 0xdf, 0xf4, 0xdf, 0x1c, 0xfe, 0xf9,
0xf7, 0x13, 0xca, 0xff, 0x03, 0x06, 0xe9, 0xf7, 0x06, 0x08, 0xd7, 0xf3,
0xed, 0x08, 0xe3, 0xfd, 0x0c, 0x11, 0x15, 0xfb, 0x15, 0x08, 0x28, 0x40,
0xe7, 0x0d, 0x08, 0xec, 0xe8, 0x16, 0x67, 0x46, 0xc8, 0x16, 0xf1, 0x02,
0x24, 0x00, 0x3a, 0x43, 0xd6, 0x12, 0xae, 0xe7, 0xf4, 0xf8, 0x3a, 0x65,
0xe4, 0x0c, 0xb2, 0xef, 0x1f, 0xe8, 0x29, 0x59, 0xf8, 0x11, 0xc4, 0xe1,
0xfe, 0xfa, 0x27, 0x43, 0xc9, 0x1e, 0xbb, 0xfb, 0xf3, 0x13, 0x15, 0x0d,
0xf1, 0x13, 0xcd, 0xf0, 0x07, 0x19, 0x07, 0x00, 0xd8, 0xeb, 0xbf, 0xf0,
0xfc, 0xf6, 0xef, 0x16, 0x01, 0x02, 0xc1, 0xdf, 0xfd, 0xe9, 0x06, 0x06,
0xf1, 0x08, 0xd7, 0xcc, 0xfb, 0x0e, 0xfc, 0x14, 0xf2, 0x1a, 0xe2, 0x0d,

0xeb, 0x09, 0x07, 0x10, 0xe6, 0x13, 0xeb, 0xf5, 0x15, 0x14, 0xeb, 0xfe,
0xf9, 0x17, 0xd2, 0xe3, 0x1e, 0xf5, 0x04, 0x0a, 0xf1, 0x0e, 0xde, 0xe7,
0x01, 0x20, 0x0c, 0xfc, 0xdc, 0xf9, 0xe5, 0xe9, 0xff, 0x1d, 0x0a, 0xfe,
0xec, 0x25, 0xaf, 0xd2, 0x01, 0x16, 0xfc, 0x17, 0xe8, 0x1e, 0xcd, 0xd9,
0xe2, 0xf1, 0xeb, 0x08, 0xff, 0x33, 0xe5, 0xfb, 0xeb, 0x04, 0xfe, 0xf7,
0xfd, 0x1f, 0xee, 0xff, 0xed, 0xf8, 0xe0, 0xff, 0xfd, 0x2b, 0x0a, 0xf5,
0x15, 0x1d, 0xf3, 0x3f, 0x16, 0xf6, 0xf2, 0xee, 0xf4, 0xef, 0xf0, 0x56,
0x0a, 0x1a, 0xbc, 0xfc, 0x2f, 0xfb, 0xf0, 0x56, 0x1e, 0x0e, 0xc6, 0xe8,
0x06, 0x0b, 0x11, 0x62, 0x3e, 0xf9, 0xb8, 0xc9, 0xed, 0xeb, 0x02, 0x63,
0x2c, 0xfd, 0xc5, 0xe9, 0x00, 0x17, 0x0f, 0x37, 0xfe, 0x20, 0xcc, 0xe0,
0xe0, 0x0e, 0xe6, 0x20, 0x0a, 0xfd, 0xdf, 0xee, 0x0b, 0x02, 0xee, 0x1f,
0xfb, 0x06, 0xd2, 0xed, 0xfe, 0xeb, 0xfc, 0x12, 0xfd, 0x14, 0x00, 0xd8,
0x08, 0xf6, 0xec, 0x17, 0xf9, 0x10, 0x00, 0xd9, 0x18, 0xf1, 0xee, 0x0f,
0xf4, 0x03, 0xee, 0xeb, 0xf0, 0xef, 0xf2, 0x06, 0x04, 0x00, 0xf4, 0x0f,
0x09, 0x06, 0xf7, 0x0b, 0xfd, 0x01, 0x03, 0x03, 0xf4, 0xf6, 0xdd, 0x14,
0x1c, 0xef, 0xf1, 0xdd, 0xf7, 0x13, 0xd9, 0x15, 0xef, 0x02, 0xd2, 0xe7,
0x05, 0x05, 0xe2, 0x09, 0xf2, 0x11, 0xf5, 0xba, 0xf0, 0x04, 0xe0, 0x01,
0x06, 0x10, 0xe6, 0xef, 0xfc, 0x12, 0xf9, 0xf4, 0x1b, 0x2f, 0xe3, 0x0f,
0xd7, 0xf6, 0x0b, 0x11, 0xf7, 0x0c, 0x00, 0x06, 0x18, 0xef, 0x06, 0x03,
0x0a, 0x09, 0xf6, 0x1a, 0x0d, 0xed, 0xfe, 0x2c, 0x43, 0xf4, 0xe5, 0xde,
0xf5, 0x02, 0x25, 0x5a, 0x49, 0xd4, 0xe6, 0x24, 0x1e, 0xf7, 0x0e, 0x5c,
0x5d, 0xf0, 0xf9, 0xe4, 0x1c, 0xeb, 0x28, 0x7f, 0x5b, 0xec, 0xfa, 0xdb,
0x0c, 0xf5, 0x20, 0x49, 0x51, 0xe1, 0xed, 0xe6, 0x0e, 0x26, 0x28, 0x33,
0x35, 0x05, 0xe1, 0xe4, 0x1f, 0xfc, 0xf9, 0x39, 0x18, 0x04, 0xed, 0xed,
0x01, 0xe7, 0xe6, 0x08, 0x09, 0x03, 0xe7, 0xf9, 0x0e, 0x06, 0xec, 0x08,
0x12, 0x1a, 0xda, 0xef, 0xdf, 0xf9, 0xe2, 0x1e, 0x1c, 0x00, 0x12, 0xd7,

0x01, 0xf7, 0x21, 0x17, 0x13, 0x19, 0xde, 0xe0, 0xec, 0x16, 0x01, 0x1b,
0x06, 0x0c, 0xf0, 0xe8, 0x18, 0x03, 0x06, 0x0e, 0x09, 0xfa, 0x03, 0xf3,
0xdd, 0x01, 0xfb, 0x0a, 0x2a, 0xf4, 0xf6, 0xda, 0xe9, 0xfe, 0xe9, 0x12,
0x19, 0xe9, 0x05, 0xdf, 0x00, 0xeb, 0xf2, 0x10, 0x0c, 0xe1, 0xcd, 0xcb,
0xf2, 0x1f, 0xd9, 0x0c, 0xfa, 0xfb, 0xe8, 0xde, 0x00, 0xfc, 0xe5, 0x00,
0x11, 0x02, 0xe6, 0x17, 0x14, 0x00, 0xf2, 0xfd, 0x00, 0xe1, 0x10, 0x24,
0x12, 0xec, 0xed, 0x1e, 0x09, 0x18, 0x03, 0x0c, 0x04, 0xf4, 0x15, 0xf0,
0x10, 0x18, 0xd6, 0x29, 0x10, 0x04, 0x1c, 0xef, 0xf0, 0x0c, 0xc7, 0x04,
0xfe, 0xeb, 0xff, 0xf5, 0xe3, 0x15, 0xfe, 0xcb, 0x10, 0xff, 0x12, 0xfb,
0xe4, 0xeb, 0xf9, 0x00, 0x02, 0xf1, 0x14, 0x13, 0x01, 0x02, 0xf9, 0x01,
0x06, 0x0c, 0xf5, 0x0a, 0x1e, 0x01, 0x19, 0x0e, 0x05, 0xf5, 0x0a, 0xff,
0xff, 0xf2, 0xfb, 0xdb, 0xf8, 0x06, 0x17, 0xf2, 0xf7, 0x0d, 0x0e, 0xf4,
0xfa, 0xf7, 0x14, 0xdb, 0xe0, 0xfd, 0x08, 0x16, 0xf7, 0x16, 0xfc, 0x09,
0x27, 0x07, 0x09, 0xfb, 0x0a, 0xfc, 0x0c, 0xe4, 0xdb, 0xee, 0xff, 0x10,
0xf3, 0x09, 0xfa, 0xf4, 0x23, 0xf3, 0xf4, 0x19, 0xff, 0xfa, 0xff, 0x19,
0xf0, 0x11, 0xed, 0xec, 0xf8, 0xf0, 0x10, 0xf3, 0xff, 0x0b, 0xf7, 0x06,
0x0b, 0x0e, 0x07, 0xe4, 0x18, 0x0a, 0x08, 0x0e, 0x02, 0x0a, 0x05, 0x19,
0x02, 0xf3, 0xfe, 0xfe, 0x0b, 0xf0, 0xfc, 0xfa, 0x05, 0xf9, 0xe2, 0xf9,
0x1b, 0xf7, 0xf0, 0x07, 0xfc, 0x12, 0xfe, 0x01, 0xfd, 0xf0, 0x04, 0xf4,
0xfd, 0x07, 0xf2, 0x04, 0x04, 0x07, 0xef, 0x0c, 0xed, 0x0e, 0xf6, 0xef,
0x08, 0x07, 0x04, 0xe9, 0xf3, 0x20, 0xda, 0x15, 0xf8, 0xff, 0xec, 0xe0,
0xf6, 0xff, 0xe9, 0x08, 0x01, 0x10, 0xf0, 0xfc, 0xe9, 0x08, 0xe8, 0xf5,
0xf8, 0xe5, 0x17, 0xe6, 0x03, 0xfc, 0x09, 0xf5, 0xdd, 0xf2, 0xff, 0x05,
0xf6, 0xf8, 0xf5, 0x07, 0xfc, 0xf1, 0x04, 0xf3, 0x13, 0xe1, 0xf0, 0xf2,
0x0a, 0xf9, 0xfd, 0x1c, 0xe0, 0x11, 0x1b, 0xe6, 0xef, 0x05, 0x05, 0x0c,
0x23, 0x10, 0x09, 0xfe, 0xf7, 0x1a, 0xf1, 0xfc, 0x11, 0x1d, 0xff, 0x03,

0x03, 0xe6, 0x07, 0x11, 0x0c, 0x0d, 0x16, 0x05, 0x05, 0x25, 0xf3, 0x10,
0x10, 0x06, 0x09, 0xe8, 0x1a, 0xf0, 0xee, 0x09, 0xff, 0x24, 0xf7, 0xfb,
0xe6, 0x06, 0xfa, 0x08, 0x03, 0x00, 0xf2, 0x04, 0xf0, 0xeb, 0x14, 0x1c,
0x03, 0x21, 0x14, 0x1d, 0xfe, 0x03, 0xf6, 0x02, 0x09, 0xff, 0x00, 0x13,
0xef, 0x10, 0x1e, 0x0b, 0x1d, 0x1c, 0xf1, 0xf6, 0xe7, 0xfd, 0x14, 0x01,
0xff, 0x13, 0xf7, 0xfc, 0x00, 0x21, 0xe3, 0xeb, 0x07, 0x0e, 0x09, 0xf1,
0xf8, 0xfd, 0x03, 0xee, 0x19, 0xfd, 0xff, 0xfb, 0xff, 0xea, 0xfb, 0x07,
0xf0, 0x0a, 0x04, 0x04, 0x0b, 0x12, 0xfe, 0x0b, 0xe0, 0xff, 0xf6, 0xe5,
0xfc, 0x11, 0xed, 0xfd, 0x15, 0x03, 0xdd, 0xdb, 0x04, 0xfe, 0xff, 0x0e,
0xff, 0xfa, 0xfb, 0xe5, 0xef, 0xf6, 0xfe, 0x22, 0x0f, 0xe8, 0xfe, 0xf4,
0xfd, 0xd9, 0x03, 0x0a, 0xdf, 0xcf, 0xf1, 0x14, 0x05, 0xfd, 0xfb, 0xf3,
0xfb, 0xfb, 0x0f, 0xf8, 0x05, 0x09, 0x03, 0xf7, 0x05, 0x05, 0x13, 0xfb,
0xeb, 0x23, 0xe7, 0x18, 0xfb, 0x00, 0xfe, 0xdd, 0xe9, 0xea, 0xd3, 0xe8,
0x1a, 0xef, 0x01, 0xf1, 0x09, 0x1d, 0xd8, 0xfc, 0xda, 0x19, 0x03, 0xec,
0xe5, 0xf3, 0xed, 0x0a, 0xf4, 0x13, 0x0b, 0xf7, 0x0c, 0x00, 0xf9, 0xea,
0xe3, 0xfe, 0xff, 0x0d, 0x0a, 0x1b, 0xd7, 0x17, 0xeb, 0xe9, 0x00, 0x0e,
0xee, 0x24, 0xef, 0x09, 0x07, 0xf0, 0xf5, 0x07, 0xf5, 0xf5, 0x10, 0x17,
0x06, 0xf7, 0xfc, 0x02, 0xfb, 0xf9, 0xe7, 0x0a, 0x26, 0xf3, 0x01, 0x01,
0x09, 0x0b, 0x02, 0x27, 0xf8, 0xee, 0xfd, 0x1c, 0xf8, 0xf2, 0x0f, 0xfc,
0x0d, 0xe0, 0xea, 0x02, 0x0b, 0x00, 0xe0, 0x08, 0xfe, 0x10, 0x04, 0xfe,
0xeb, 0x13, 0x01, 0x0c, 0x0e, 0xed, 0x09, 0x01, 0x0c, 0xe3, 0x10, 0xdf,
0xd1, 0x14, 0xf3, 0xef, 0x09, 0xf0, 0xee, 0xe5, 0x11, 0xf4, 0xf6, 0x00,
0xe8, 0x20, 0x0a, 0xfc, 0xea, 0xf7, 0x02, 0x16, 0xe7, 0xf3, 0x0d, 0xe4,
0x04, 0xe6, 0xef, 0xf8, 0x0f, 0x23, 0x02, 0xe0, 0x01, 0x01, 0x01, 0x05,
0xf5, 0x0d, 0xf5, 0xf5, 0xe1, 0xff, 0x04, 0x00, 0xf4, 0x0d, 0xee, 0xf1,
0xef, 0xf7, 0x0b, 0xff, 0x1b, 0xec, 0x05, 0xe7, 0xf3, 0x13, 0x12, 0xf2,

0xf3, 0xfc, 0xea, 0x06, 0xfe, 0x13, 0x12, 0xdb, 0x11, 0xe2, 0xfc, 0x0d,
0x1c, 0xe8, 0x1d, 0xfc, 0xf2, 0xe2, 0x13, 0x1d, 0xda, 0xf6, 0x1c, 0x18,
0x1e, 0xf4, 0xfa, 0x03, 0xdc, 0x0f, 0xff, 0xff, 0x18, 0x0b, 0xed, 0xf1,
0xf8, 0x02, 0xf4, 0x10, 0xf9, 0xeb, 0x0b, 0x0e, 0x0f, 0x01, 0x02, 0x1b,
0x06, 0x10, 0x00, 0xe7, 0x23, 0x0d, 0xf6, 0x11, 0x08, 0xf5, 0x0f, 0x05,
0x13, 0xf7, 0x01, 0x01, 0x0c, 0xf6, 0xf9, 0xf0, 0x29, 0x01, 0xe9, 0x11,
0x02, 0xfa, 0xeb, 0x16, 0x0e, 0x10, 0x09, 0x0e, 0x1c, 0x0a, 0xe3, 0xd3,
0x01, 0xe3, 0x00, 0x06, 0xe2, 0xe9, 0x19, 0xef, 0x12, 0xf3, 0xfc, 0x02,
0x0b, 0x0c, 0x0d, 0xed, 0xfd, 0xf6, 0xf9, 0xe9, 0xf2, 0x28, 0xfe, 0x03,
0xec, 0x03, 0x00, 0xf8, 0xde, 0x0d, 0x25, 0x07, 0x1a, 0xe7, 0xfd, 0x29,
0xd8, 0xf7, 0xfb, 0xde, 0x0c, 0x08, 0x06, 0x22, 0xee, 0x1d, 0x05, 0x07,
0xf0, 0xfb, 0xfe, 0x07, 0xf1, 0x04, 0xe9, 0x01, 0xfc, 0xf1, 0x00, 0xeb,
0xe3, 0x08, 0xec, 0xfe, 0x04, 0xeb, 0xfc, 0x01, 0xf6, 0x0e, 0xdf, 0xf8,
0x12, 0xe3, 0x16, 0xdc, 0x21, 0x0a, 0xe6, 0x06, 0xe5, 0x10, 0x07, 0xf7,
0x1e, 0xde, 0xe3, 0x07, 0x16, 0xed, 0x23, 0xf2, 0x12, 0x0d, 0xe9, 0xf9,
0xe8, 0xfe, 0x0e, 0x02, 0x18, 0x0a, 0xea, 0xec, 0xfb, 0xfe, 0x0c, 0x1b,
0x19, 0x20, 0xfa, 0x07, 0xe5, 0x0c, 0x04, 0x27, 0xdb, 0xe6, 0xfe, 0x0d,
0x0a, 0x0a, 0xfe, 0x39, 0xdd, 0xde, 0x05, 0xec, 0x09, 0x05, 0x0a, 0x2c,
0xf4, 0x02, 0x1f, 0xd3, 0x24, 0xee, 0x0f, 0x3c, 0xf5, 0xfd, 0xf8, 0xf8,
0x12, 0xf5, 0xf3, 0x19, 0xf9, 0xda, 0xf6, 0x0a, 0x0a, 0xf4, 0x09, 0x0f,
0xfc, 0x00, 0x01, 0x01, 0xf3, 0xf8, 0x05, 0xf3, 0x0c, 0x19, 0x0e, 0xfd,
0xfa, 0xe1, 0xfc, 0x0c, 0x03, 0xfb, 0x1b, 0x06, 0xcc, 0xe4, 0x08, 0xf9,
0x10, 0xe9, 0x06, 0x00, 0x17, 0xe8, 0x0d, 0x12, 0xca, 0xf5, 0x23, 0xe4,
0x21, 0xf6, 0x19, 0x33, 0xdd, 0xfa, 0x0c, 0x01, 0x14, 0x07, 0x00, 0x34,
0xda, 0x05, 0x07, 0x01, 0x07, 0xe4, 0x06, 0x24, 0x02, 0xff, 0xf0, 0x09,
0xfc, 0xf4, 0x03, 0x06, 0xee, 0x08, 0xe2, 0x1d, 0xfa, 0x0c, 0xfc, 0x02,

0x03, 0xe5, 0xf0, 0xe2, 0x0a, 0x18, 0x12, 0x0c, 0x1e, 0x20, 0xed, 0x20,
0xe4, 0x01, 0x2a, 0x09, 0x0d, 0x0e, 0xd0, 0xf4, 0xdd, 0xfd, 0x2b, 0xf2,
0x08, 0x0c, 0xf8, 0xf7, 0xfc, 0xf9, 0x15, 0xef, 0x19, 0x1c, 0x01, 0xff,
0xe2, 0x01, 0xf3, 0x30, 0x0e, 0xfb, 0x15, 0xe8, 0x1c, 0x00, 0xfa, 0x16,
0xef, 0xea, 0xfb, 0x05, 0xf0, 0x0e, 0x02, 0x13, 0xf4, 0x01, 0x03, 0xe5,
0x29, 0x07, 0x09, 0x24, 0xf9, 0xe3, 0xf8, 0xde, 0x2d, 0xf4, 0xf5, 0x40,
0xed, 0xdf, 0x07, 0xef, 0x0f, 0x0a, 0x0b, 0x32, 0x0d, 0xe8, 0x00, 0xe6,
0xf6, 0xfc, 0xfd, 0x19, 0x11, 0x09, 0xf3, 0x03, 0xea, 0xf1, 0xfb, 0x02,
0xfd, 0x06, 0xff, 0xfe, 0x09, 0xec, 0x06, 0x0c, 0x15, 0xf9, 0x06, 0xd7,
0xe3, 0xf7, 0xed, 0x01, 0x03, 0xfd, 0x14, 0x01, 0x0e, 0xe0, 0x37, 0x0d,
0xd2, 0x18, 0x2f, 0xea, 0x12, 0x0d, 0x05, 0x3a, 0xd5, 0x07, 0x1e, 0xf2,
0x21, 0x11, 0xf9, 0x36, 0xd3, 0xf5, 0x12, 0xf6, 0xfb, 0xf6, 0x06, 0x0f,
0xde, 0xf9, 0x06, 0x09, 0xdf, 0xff, 0x0b, 0xf3, 0xf5, 0x01, 0xf1, 0xea,
0xf2, 0x02, 0x12, 0xfc, 0x0e, 0xee, 0xf8, 0xeb, 0x00, 0xef, 0x21, 0x0f,
0x09, 0xef, 0xeb, 0x1e, 0xef, 0xf2, 0x26, 0xf9, 0x17, 0xf1, 0xf1, 0xf0,
0x0c, 0x10, 0x1d, 0xff, 0x1d, 0x06, 0x03, 0xf6, 0xfb, 0x14, 0x1b, 0x03,
0x22, 0xfd, 0xec, 0x03, 0xfa, 0xf8, 0x01, 0x2b, 0x1e, 0x1b, 0x09, 0x09,
0x07, 0xff, 0xf0, 0x20, 0xee, 0x14, 0xfb, 0xf6, 0xf8, 0x11, 0xd9, 0x29,
0xf4, 0xfa, 0x07, 0xef, 0x20, 0xf9, 0xf2, 0x30, 0xee, 0xf0, 0xf3, 0xd6,
0x0d, 0xfe, 0x03, 0x36, 0xf5, 0xd7, 0x01, 0xe6, 0x04, 0xf0, 0x05, 0x1f,
0x0f, 0xdd, 0xff, 0xf8, 0x1f, 0xf2, 0x04, 0x37, 0xfa, 0x00, 0xfd, 0xf8,
0x10, 0xe1, 0xfb, 0x0d, 0xed, 0xf6, 0xe2, 0xfe, 0x08, 0xfe, 0x07, 0x08,
0x08, 0x11, 0x0a, 0xf0, 0xf8, 0xf5, 0x04, 0xea, 0x08, 0x12, 0x06, 0x0d,
0x0f, 0x10, 0x40, 0x28, 0xc0, 0xfb, 0x3f, 0x08, 0x1d, 0x09, 0x1b, 0x3d,
0xee, 0xf4, 0x29, 0x13, 0x20, 0xfc, 0x11, 0x4c, 0xdb, 0x02, 0x15, 0x05,
0xec, 0xeb, 0x0a, 0x22, 0xe7, 0x00, 0x02, 0x01, 0xd4, 0xea, 0x0a, 0xf3,

0xe3, 0xf8, 0xf5, 0xfa, 0x01, 0x0d, 0x19, 0x06, 0x24, 0x13, 0x02, 0xf5,
0xf1, 0xf1, 0x1b, 0x0f, 0x19, 0x04, 0xe3, 0xf9, 0xe7, 0x02, 0x29, 0xfc,
0x29, 0xec, 0xe9, 0x04, 0xdc, 0x22, 0x1d, 0xfd, 0x1f, 0x01, 0xec, 0xe8,
0xf5, 0x14, 0x1b, 0x19, 0x06, 0x0e, 0x02, 0x0d, 0xf9, 0x06, 0xfc, 0x15,
0x07, 0xfa, 0x0c, 0xe1, 0x18, 0x1a, 0xe8, 0x1b, 0xe9, 0xef, 0x0a, 0x18,
0xfc, 0x05, 0xf9, 0x14, 0xdc, 0x04, 0x01, 0xff, 0x07, 0xfd, 0xf0, 0x2c,
0xf2, 0xec, 0x0e, 0xe7, 0x1a, 0x05, 0xe8, 0x35, 0x13, 0x09, 0xf9, 0x07,
0xfe, 0xfa, 0x0d, 0x40, 0x0c, 0xea, 0xf4, 0x04, 0x01, 0x11, 0xfc, 0x23,
0xeb, 0xf4, 0xe9, 0x04, 0xeb, 0xe7, 0x07, 0x09, 0xfb, 0xf1, 0xf6, 0xfd,
0x02, 0xfa, 0x02, 0xff, 0x00, 0xff, 0xf1, 0xf1, 0x1a, 0xe9, 0x10, 0xe3,
0x0b, 0x0c, 0x08, 0x04, 0x1b, 0x0a, 0x2b, 0x10, 0xe1, 0x01, 0x1f, 0x06,
0x04, 0xec, 0x19, 0x49, 0xee, 0xf8, 0x22, 0x0c, 0x20, 0x02, 0x07, 0x31,
0xe7, 0xff, 0x0f, 0xf0, 0xfd, 0xea, 0x13, 0x26, 0xce, 0xfa, 0xff, 0xee,
0xe9, 0xfe, 0x15, 0x08, 0x04, 0x05, 0x0d, 0xfa, 0xdd, 0xf8, 0x07, 0x0b,
0x33, 0xef, 0xec, 0xf9, 0xd9, 0xe6, 0x1d, 0x10, 0x41, 0xf6, 0xdf, 0x11,
0xe3, 0x14, 0x1d, 0xfb, 0x2b, 0x15, 0xdc, 0x09, 0xf6, 0x05, 0x16, 0x00,
0x1c, 0x27, 0xe4, 0xfc, 0xf7, 0x16, 0x08, 0x08, 0x2f, 0xdd, 0xf8, 0xfa,
0xe9, 0x0e, 0x0b, 0x0b, 0x02, 0x12, 0x02, 0xfd, 0x19, 0x03, 0xeb, 0x11,
0xf4, 0x09, 0x09, 0x15, 0x12, 0x0d, 0xef, 0x1c, 0xe4, 0xfe, 0x17, 0x0c,
0x09, 0x04, 0xea, 0x2f, 0xf2, 0x1e, 0x02, 0xfb, 0xfe, 0xe3, 0x00, 0x2e,
0x04, 0xf9, 0x0c, 0x05, 0x27, 0x0c, 0x07, 0x2d, 0xf7, 0x0b, 0xfb, 0xf9,
0x1c, 0xdf, 0x11, 0x36, 0x05, 0xf2, 0x02, 0xf8, 0x0b, 0x07, 0x05, 0xfb,
0xfc, 0x0e, 0x13, 0xfa, 0xfb, 0x09, 0xf5, 0xfd, 0x06, 0x15, 0xf9, 0x03,
0x18, 0xfd, 0x1a, 0x0a, 0x03, 0xe2, 0xfb, 0x00, 0x1e, 0xfe, 0x4f, 0x27,
0xe1, 0xf7, 0x31, 0xf0, 0x1b, 0xec, 0x07, 0x5f, 0xe2, 0xf8, 0x40, 0x05,
0x17, 0x24, 0x0c, 0x3c, 0xf3, 0x10, 0x13, 0xf8, 0x0b, 0xf3, 0xf9, 0x36,

0xe1, 0xf3, 0xf4, 0xe8, 0xef, 0xf8, 0xfc, 0xeb, 0xe3, 0xfb, 0xf0, 0xee,
0xdb, 0x06, 0x0c, 0x11, 0x1e, 0x10, 0xe2, 0xe9, 0xeb, 0x0d, 0x34, 0x0f,
0x43, 0xd9, 0xef, 0x08, 0xec, 0x05, 0x1d, 0x02, 0x33, 0xef, 0xf4, 0xf7,
0xe6, 0xf9, 0x22, 0x07, 0x04, 0x06, 0xe9, 0x02, 0xf0, 0xfc, 0x24, 0x20,
0x24, 0x17, 0xe6, 0x0f, 0x05, 0xf6, 0xfc, 0x1f, 0xf2, 0x01, 0x0d, 0xe7,
0xff, 0x1d, 0xf0, 0xfa, 0xd0, 0x00, 0xff, 0x0e, 0x23, 0xf9, 0xf3, 0x11,
0xde, 0x0d, 0x05, 0x04, 0x0b, 0x0b, 0xfb, 0x26, 0x0d, 0x0d, 0xff, 0xe8,
0x16, 0xe8, 0x0b, 0x3c, 0x18, 0xe4, 0x04, 0xff, 0xfa, 0xf3, 0xff, 0x40,
0xee, 0x06, 0xfc, 0x0d, 0x00, 0xf7, 0x13, 0x3f, 0xf7, 0x13, 0x06, 0x08,
0xf9, 0x13, 0xf2, 0x19, 0xfd, 0xf9, 0xf3, 0xe6, 0xfc, 0x07, 0xf6, 0xfd,
0x0a, 0x22, 0x00, 0x01, 0x19, 0xff, 0xe7, 0xff, 0x08, 0xfd, 0x03, 0xfd,
0x1f, 0xe7, 0x28, 0x08, 0xde, 0xf3, 0x43, 0xf6, 0x0c, 0xfe, 0x1e, 0x52,
0xf2, 0x04, 0x17, 0xf2, 0x08, 0x0d, 0x04, 0x38, 0xde, 0x0c, 0x10, 0xef,
0xdf, 0x0f, 0x01, 0x24, 0xde, 0xe1, 0x0d, 0xfd, 0xd4, 0xf6, 0x12, 0x0e,
0xed, 0x01, 0xf0, 0xf3, 0xfd, 0xff, 0x18, 0xf3, 0x36, 0xda, 0xf6, 0xef,
0xe8, 0xef, 0x37, 0x27, 0x4e, 0xf8, 0xf4, 0xff, 0xe5, 0xf3, 0x32, 0x0b,
0x36, 0x08, 0xe9, 0xf6, 0xe2, 0x13, 0x21, 0xfe, 0x12, 0xed, 0xdd, 0xfb,
0xf8, 0x05, 0x0f, 0x03, 0x1c, 0x04, 0xfc, 0xf2, 0x23, 0x0e, 0x03, 0xfc,
0xf9, 0x18, 0xf7, 0x01, 0x1b, 0x03, 0xf5, 0xfd, 0xde, 0xf3, 0x19, 0xfc,
0x11, 0x02, 0xe7, 0x13, 0xde, 0xd8, 0xf2, 0x05, 0x28, 0x02, 0x02, 0x27,
0x07, 0x08, 0xff, 0x07, 0x27, 0x0e, 0x19, 0x40, 0xfb, 0x02, 0x0c, 0xf6,
0x0d, 0x07, 0x0f, 0x47, 0xf8, 0x05, 0x0e, 0xfd, 0x03, 0x1e, 0x07, 0x32,
0xe7, 0xf6, 0x24, 0x01, 0x01, 0x02, 0x0a, 0xff, 0xf6, 0x26, 0x15, 0xf0,
0x04, 0x13, 0x03, 0xfa, 0xfe, 0xf6, 0xf1, 0x09, 0x2a, 0xe6, 0xea, 0xf6,
0x17, 0x13, 0xeb, 0xff, 0x15, 0xeb, 0x23, 0x06, 0xc8, 0xf6, 0x33, 0xeb,
0xf4, 0xe7, 0x12, 0x2a, 0xe3, 0xe6, 0x32, 0xfa, 0x16, 0x15, 0x17, 0x40,

0xf1, 0x08, 0x1a, 0xf3, 0xf6, 0x0c, 0x0c, 0x11, 0xd0, 0x22, 0x02, 0xee,
0xea, 0xf4, 0xf8, 0xf9, 0x13, 0x10, 0x17, 0xf5, 0xf1, 0x0a, 0x0e, 0xfd,
0x32, 0xda, 0xf1, 0xe2, 0xdb, 0xf2, 0x34, 0x1f, 0x53, 0xfc, 0xe4, 0xf2,
0xf6, 0xf2, 0x1d, 0x04, 0x4a, 0xec, 0xee, 0x06, 0xdf, 0x01, 0x1a, 0x04,
0x27, 0xfc, 0xe6, 0xfd, 0xd9, 0xfd, 0x0e, 0x00, 0x0c, 0x16, 0xf3, 0x03,
0xf7, 0xfc, 0x0e, 0x0f, 0x09, 0x06, 0x06, 0x04, 0x08, 0x02, 0xed, 0xf5,
0xe4, 0xe6, 0x07, 0x06, 0x03, 0x18, 0xea, 0x13, 0xe2, 0xfa, 0x10, 0xf2,
0x02, 0xec, 0x03, 0x3c, 0xf6, 0xf6, 0x0a, 0x10, 0x09, 0xf8, 0x15, 0x24,
0xfd, 0x0d, 0x09, 0x01, 0x00, 0xff, 0x00, 0x1a, 0xf0, 0xee, 0x08, 0x03,
0x1d, 0x05, 0x16, 0x46, 0xe6, 0xf8, 0x08, 0x00, 0x09, 0x09, 0xff, 0x01,
0xfc, 0x20, 0xfc, 0xec, 0x05, 0x1b, 0x03, 0xf1, 0x12, 0xe4, 0xfa, 0x24,
0x1c, 0xf5, 0xf2, 0x05, 0x11, 0xe7, 0xfa, 0x02, 0x20, 0xea, 0x31, 0x10,
0xcf, 0xd8, 0x33, 0xee, 0xff, 0x09, 0x20, 0x3f, 0xe2, 0x0a, 0x29, 0xee,
0x3a, 0xf2, 0x1e, 0x39, 0x02, 0x1e, 0xfe, 0xf2, 0xef, 0xe2, 0x0d, 0x0f,
0xf1, 0x19, 0x02, 0xe7, 0xec, 0xff, 0xfe, 0xe4, 0xfe, 0xfb, 0x02, 0xf6,
0xf1, 0xf4, 0x07, 0x1a, 0x2a, 0xf9, 0x06, 0xf9, 0xda, 0xf4, 0x22, 0x02,
0x4f, 0x0a, 0xf3, 0xfc, 0xf3, 0xf6, 0x25, 0x0a, 0x28, 0x01, 0xf7, 0x09,
0xe6, 0x05, 0x28, 0xf7, 0x1e, 0xf2, 0xee, 0x13, 0xee, 0x05, 0x0f, 0x0a,
0x09, 0xe8, 0xe8, 0x0e, 0x05, 0x12, 0x0f, 0x15, 0x02, 0xec, 0xf8, 0x02,
0xf7, 0x05, 0xf8, 0xff, 0xdc, 0x00, 0x01, 0x00, 0x12, 0x17, 0xec, 0x19,
0xfa, 0x09, 0xfa, 0xf3, 0x1d, 0x0b, 0x07, 0x25, 0xea, 0x0c, 0xf5, 0xfa,
0x04, 0xf7, 0xfe, 0x33, 0xfe, 0x14, 0xef, 0x04, 0xf0, 0x00, 0x00, 0x3a,
0xea, 0xfa, 0x10, 0x01, 0xe4, 0x00, 0xff, 0x23, 0xe9, 0x26, 0x15, 0x10,
0x04, 0x14, 0x0d, 0x08, 0xf8, 0xfd, 0x10, 0xfb, 0x00, 0x21, 0x06, 0xfa,
0x0f, 0x08, 0xf1, 0x09, 0x28, 0xf0, 0xd8, 0x0d, 0x08, 0x09, 0x02, 0xfb,
0x12, 0x03, 0x0e, 0xfb, 0xce, 0xf0, 0x39, 0xe5, 0x09, 0xf6, 0x1f, 0x35,

0xdd, 0x1c, 0x25, 0xef, 0x17, 0x0c, 0xf6, 0x3e, 0xf0, 0x21, 0x08, 0xff,
0xd7, 0xfc, 0xfd, 0x1f, 0xe5, 0x18, 0x12, 0xe9, 0xf5, 0xe9, 0x12, 0xf6,
0x02, 0x13, 0xf4, 0x0a, 0xfd, 0x03, 0x09, 0x08, 0x2f, 0x07, 0xee, 0xfd,
0xd7, 0x00, 0x2b, 0x29, 0x3b, 0xdb, 0xde, 0xf1, 0xe1, 0xf7, 0x47, 0x12,
0x35, 0x0c, 0xe4, 0x09, 0xef, 0x17, 0x2b, 0xea, 0x2d, 0xf8, 0xe8, 0x18,
0xef, 0x03, 0x11, 0x0a, 0x10, 0xff, 0xe8, 0x07, 0x0c, 0x07, 0x03, 0x18,
0x05, 0x08, 0xf8, 0xf8, 0x06, 0x18, 0xe9, 0xf9, 0xe0, 0x0f, 0x0d, 0x18,
0x04, 0x01, 0xf0, 0x1c, 0xf6, 0x14, 0xfd, 0x12, 0x0c, 0x0c, 0x02, 0x34,
0xf6, 0xe6, 0xfd, 0xf9, 0xf9, 0xfd, 0x00, 0x2a, 0xfc, 0xf9, 0xff, 0x0a,
0xfe, 0x1b, 0xf5, 0x34, 0xdc, 0xf9, 0x15, 0x13, 0xe7, 0x1b, 0xf7, 0x25,
0xfd, 0x09, 0x08, 0x0a, 0xf0, 0x17, 0x0f, 0x04, 0xf4, 0xe9, 0x06, 0x07,
0xf5, 0x02, 0xfc, 0xf5, 0x09, 0xee, 0xf1, 0x07, 0x38, 0x03, 0x05, 0x0f,
0x16, 0x0f, 0xed, 0xff, 0x21, 0xf8, 0x34, 0x07, 0xd1, 0xf9, 0x27, 0x00,
0x0c, 0x21, 0x18, 0x42, 0xe6, 0x02, 0x1a, 0xf1, 0x2f, 0xf1, 0x0e, 0x3b,
0xee, 0xf8, 0x08, 0xea, 0xfe, 0xf9, 0x03, 0x18, 0xf5, 0xf8, 0x0d, 0xeb,
0x01, 0x10, 0x09, 0x02, 0x15, 0xfb, 0xf1, 0x0b, 0xf2, 0x06, 0x08, 0x09,
0x2f, 0x19, 0x02, 0xfe, 0xe4, 0x06, 0x1f, 0x17, 0x49, 0xf2, 0xe2, 0x02,
0xef, 0x04, 0x26, 0x16, 0x3f, 0x08, 0xf1, 0x0a, 0xfd, 0xf9, 0x28, 0x01,
0x15, 0x0b, 0xf9, 0x10, 0xdc, 0x02, 0x20, 0xf7, 0x16, 0xe6, 0x09, 0x03,
0xf1, 0xf5, 0x12, 0x1c, 0xfb, 0x2a, 0x08, 0xfa, 0x0a, 0x16, 0xf6, 0x15,
0xf0, 0x06, 0x11, 0xfd, 0x0e, 0xf9, 0xf6, 0x12, 0xed, 0xf3, 0xfd, 0x1f,
0x0b, 0xfa, 0x08, 0x30, 0xf8, 0xff, 0x0b, 0xeb, 0x10, 0xff, 0x07, 0x22,
0x0d, 0x07, 0x09, 0x03, 0xf6, 0xf8, 0xfc, 0x26, 0xf8, 0xee, 0x11, 0x02,
0x03, 0x0a, 0xef, 0x38, 0xfe, 0x13, 0x1b, 0x09, 0xfe, 0x06, 0x05, 0xf3,
0x04, 0xdf, 0xfc, 0x00, 0xe7, 0x15, 0xec, 0xf1, 0xf8, 0xfc, 0xed, 0x05,
0x0e, 0xf3, 0x15, 0x09, 0x01, 0x0d, 0xfd, 0x00, 0x24, 0xe2, 0x31, 0x13,

0xd5, 0x1b, 0x2b, 0xe8, 0x03, 0x08, 0x1d, 0x33, 0xdc, 0xfd, 0x24, 0xe4,
0x20, 0xfa, 0x07, 0x33, 0x01, 0x12, 0x06, 0xf5, 0xef, 0xf7, 0xfa, 0x13,
0x01, 0xec, 0xee, 0xe0, 0xfd, 0x0d, 0xff, 0x09, 0xf6, 0x00, 0xed, 0x07,
0xea, 0x0e, 0xff, 0x0e, 0x26, 0xfc, 0xf0, 0xe7, 0xe7, 0xfe, 0x30, 0xff,
0x24, 0x04, 0x06, 0xf4, 0xf5, 0xf8, 0x23, 0x0e, 0x3d, 0xf2, 0xfd, 0x04,
0xe8, 0xfb, 0x23, 0xfe, 0x33, 0xe1, 0x01, 0xfd, 0xdc, 0xfb, 0x0e, 0xfa,
0x22, 0xfb, 0x11, 0xfa, 0xff, 0x08, 0x21, 0x30, 0x13, 0x03, 0xf2, 0x03,
0xf8, 0x0f, 0xec, 0x0d, 0xef, 0x0f, 0x10, 0x10, 0x0f, 0xf6, 0xf9, 0x1e,
0xf7, 0xe5, 0x08, 0xfa, 0x09, 0xff, 0x00, 0x15, 0x02, 0x00, 0x08, 0xfe,
0xfb, 0x0e, 0x15, 0x28, 0xfa, 0xfb, 0x13, 0x06, 0xfb, 0x05, 0xf6, 0x11,
0xf6, 0x0b, 0x06, 0x15, 0xe1, 0x00, 0xe9, 0x0f, 0xe1, 0x1d, 0x18, 0xfd,
0x0b, 0x0f, 0xff, 0xf2, 0xf5, 0xfd, 0x14, 0xff, 0xf4, 0xfe, 0xe2, 0xf8,
0x14, 0x0b, 0xeb, 0x07, 0x35, 0xe2, 0xeb, 0x0b, 0x04, 0x22, 0xfe, 0x0e,
0x1d, 0xf2, 0x24, 0x11, 0xcc, 0xec, 0x25, 0xf7, 0xff, 0xf9, 0x06, 0x29,
0xe4, 0x07, 0x1c, 0xdb, 0xf8, 0x1d, 0xfa, 0x44, 0xf2, 0x01, 0x0f, 0xe6,
0x11, 0x03, 0xee, 0x17, 0x06, 0xe0, 0x0c, 0xd8, 0xe9, 0xfd, 0x11, 0xfe,
0x07, 0xdd, 0xea, 0xff, 0xde, 0xdd, 0x0a, 0x09, 0x30, 0xf2, 0x01, 0xe4,
0xe0, 0xeb, 0x2d, 0x12, 0x2d, 0xeb, 0xfc, 0xf0, 0xe8, 0xf9, 0x1f, 0x08,
0x3f, 0xeb, 0x0e, 0x13, 0xf9, 0x0c, 0x1c, 0x02, 0x25, 0xec, 0xf6, 0x05,
0xf3, 0xf4, 0x18, 0x08, 0x12, 0xe9, 0xfb, 0xfd, 0xf9, 0x08, 0x13, 0x1c,
0x08, 0xec, 0xfe, 0x02, 0xf1, 0x19, 0xf3, 0x1d, 0xf1, 0x07, 0x11, 0x12,
0xfa, 0xf2, 0xf6, 0x0d, 0xff, 0x17, 0x0a, 0xfb, 0x1f, 0xf8, 0x11, 0x24,
0xf6, 0xfc, 0xfe, 0x07, 0xed, 0x05, 0x1c, 0x21, 0xfe, 0xfe, 0x16, 0x0d,
0x08, 0x0f, 0x09, 0x33, 0xf4, 0x1f, 0x14, 0x0c, 0xfe, 0xf5, 0xeb, 0x2a,
0xee, 0xf3, 0x12, 0x19, 0xec, 0x01, 0x06, 0xf7, 0x05, 0x22, 0x0b, 0xeb,
0xeb, 0x06, 0xe1, 0xf5, 0x0d, 0xee, 0xfb, 0x0a, 0x31, 0xff, 0xe3, 0xea,

0x18, 0x09, 0xe3, 0x07, 0x1a, 0xf8, 0x15, 0xfc, 0xcc, 0xf2, 0x2a, 0xe5,
0x01, 0xea, 0x10, 0x1f, 0xd9, 0x02, 0x13, 0xf6, 0x16, 0x01, 0x0e, 0x3c,
0x02, 0x17, 0x04, 0xf1, 0xf7, 0x02, 0x07, 0x0c, 0x02, 0x1f, 0xf4, 0xe6,
0xf0, 0xe9, 0x05, 0xf4, 0xfd, 0xe4, 0xf7, 0xe9, 0xfc, 0xef, 0x06, 0x02,
0x26, 0xf1, 0xf1, 0xeb, 0xe9, 0xe6, 0x30, 0x1c, 0x38, 0x0f, 0x03, 0xf1,
0x10, 0x04, 0x30, 0x19, 0x1f, 0xfb, 0xfc, 0x05, 0xe2, 0xfe, 0x18, 0xf2,
0x1c, 0xf2, 0xf5, 0x0e, 0xf2, 0x05, 0x1d, 0x28, 0x12, 0xf0, 0xf0, 0x0f,
0x0a, 0x03, 0x1a, 0x1a, 0xf3, 0x08, 0x13, 0xef, 0xf5, 0x1c, 0x06, 0x00,
0xee, 0x12, 0x1d, 0x03, 0x18, 0x06, 0x0a, 0x0e, 0xf0, 0xeb, 0xfa, 0x0d,
0x08, 0xff, 0x06, 0x24, 0x0f, 0x03, 0x0a, 0x0f, 0x0e, 0xff, 0x08, 0x33,
0xfc, 0x00, 0x0e, 0xfb, 0xfb, 0x05, 0x07, 0x19, 0xe8, 0xe7, 0x12, 0x11,
0x15, 0xf7, 0x0c, 0x1a, 0xf6, 0x28, 0x08, 0xeb, 0xf2, 0x25, 0xee, 0x01,
0x03, 0xec, 0xed, 0xfa, 0xf0, 0xf2, 0xef, 0xf1, 0x02, 0x23, 0xef, 0x01,
0x41, 0xfa, 0xf4, 0xf4, 0x15, 0xf5, 0xf5, 0xf9, 0x28, 0xde, 0x20, 0xf6,
0xc7, 0xde, 0x21, 0xe4, 0xfe, 0xec, 0x0d, 0x2c, 0xee, 0x24, 0x10, 0xf0,
0x1d, 0x12, 0x0e, 0x2b, 0x06, 0xf8, 0xfd, 0x01, 0x08, 0xef, 0xfd, 0x0f,
0xeb, 0xed, 0xe1, 0xdf, 0xf1, 0xe5, 0x16, 0xe3, 0x08, 0xfc, 0xf6, 0xf6,
0xd8, 0xf0, 0x23, 0xfc, 0x2b, 0xf5, 0xff, 0xe7, 0xf4, 0xe9, 0x29, 0x09,
0x2b, 0x0c, 0xff, 0x08, 0x0b, 0xed, 0x29, 0x14, 0x3c, 0xf5, 0xeb, 0x18,
0xf6, 0x10, 0x22, 0xf9, 0x17, 0x23, 0x02, 0x0c, 0xf6, 0xfa, 0x2f, 0xfe,
0x1e, 0xeb, 0xfd, 0x03, 0xf0, 0x07, 0x1c, 0x09, 0xfa, 0xe1, 0x0d, 0x0f,
0x18, 0x03, 0xfe, 0xf0, 0xec, 0x0b, 0x10, 0x02, 0x14, 0x06, 0xef, 0xf7,
0xea, 0x0b, 0x05, 0xfe, 0x1f, 0x06, 0x0e, 0x07, 0x00, 0xe1, 0x01, 0x01,
0x07, 0x05, 0x09, 0xf7, 0xef, 0x15, 0xf7, 0x12, 0x05, 0x03, 0x04, 0x1d,
0x04, 0x10, 0x12, 0x06, 0x05, 0x00, 0x08, 0x18, 0xd6, 0xf2, 0xfa, 0x07,
0xf8, 0x12, 0x07, 0xfd, 0xdd, 0x00, 0x04, 0xfb, 0xf8, 0x09, 0xf3, 0x09,

0xfb, 0xf0, 0xe8, 0x09, 0x27, 0xf5, 0xf8, 0x06, 0x01, 0x02, 0x0e, 0xf6,
0x1f, 0xfa, 0x29, 0xf8, 0xd6, 0x01, 0x22, 0xf8, 0x1d, 0xe3, 0x1a, 0x39,
0x0a, 0x0d, 0x19, 0xf5, 0x12, 0xfb, 0x1d, 0x2a, 0x03, 0xf6, 0x0c, 0xf2,
0xfd, 0xec, 0x18, 0x13, 0xfe, 0x1a, 0xe8, 0xdd, 0x01, 0xf8, 0x30, 0x01,
0xf8, 0xfe, 0xe4, 0xe7, 0xff, 0xeb, 0x23, 0xfa, 0x2c, 0xf0, 0xfc, 0xe7,
0x0a, 0xf8, 0x18, 0x10, 0x23, 0x01, 0xfa, 0xe8, 0xf1, 0xfa, 0x1d, 0x0e,
0x17, 0xe7, 0xe4, 0xf5, 0xf9, 0x0c, 0x17, 0x0c, 0x13, 0xe8, 0xe1, 0x17,
0x19, 0x05, 0x0b, 0x0f, 0x23, 0xed, 0xff, 0xfe, 0xe0, 0x14, 0x16, 0x00,
0x0d, 0x1c, 0x0b, 0xf5, 0xfb, 0x18, 0xee, 0xff, 0xff, 0xf3, 0x18, 0x0c,
0x05, 0xfa, 0xf6, 0xfe, 0xfe, 0xf8, 0xf8, 0x09, 0xef, 0xf8, 0x0e, 0xf0,
0x00, 0xf8, 0x0c, 0xf8, 0xf6, 0x07, 0x16, 0x11, 0xf8, 0xea, 0xff, 0xff,
0x01, 0x20, 0x07, 0x08, 0xfd, 0x1c, 0xfc, 0x06, 0xed, 0x0d, 0x08, 0x15,
0xf0, 0x25, 0x01, 0x1b, 0x00, 0x02, 0xfe, 0x01, 0x05, 0x01, 0xfd, 0xf1,
0xe5, 0x0c, 0xe4, 0xe1, 0xf0, 0xfa, 0xee, 0x0e, 0x35, 0xee, 0x15, 0xef,
0x0a, 0xf9, 0x01, 0xf5, 0x1f, 0x05, 0x1f, 0x0d, 0xe1, 0xf4, 0xff, 0xf5,
0x23, 0x02, 0x18, 0x30, 0xfc, 0xf0, 0x0d, 0x04, 0x0d, 0x06, 0x29, 0x1d,
0xf9, 0x08, 0x06, 0xe5, 0x13, 0xfd, 0x0d, 0x26, 0xef, 0x09, 0xdc, 0xf2,
0x05, 0xdf, 0x0c, 0xf6, 0xf3, 0xd9, 0xf8, 0x08, 0xef, 0xeb, 0x0f, 0xf9,
0x3a, 0x03, 0xff, 0xe0, 0xf7, 0xf0, 0x15, 0x12, 0x41, 0x0b, 0xf1, 0x04,
0x04, 0xe2, 0x0e, 0x0b, 0x2c, 0x03, 0xea, 0x02, 0xfb, 0xe7, 0x08, 0xe9,
0x22, 0xf3, 0xf2, 0x1c, 0xfa, 0xf3, 0x11, 0x04, 0x1f, 0xf5, 0x02, 0x0f,
0x1a, 0x1f, 0x24, 0x0b, 0x06, 0x1f, 0xf3, 0x06, 0x00, 0x02, 0xe8, 0xf6,
0xf4, 0xe8, 0x07, 0x2e, 0xfb, 0xf8, 0x10, 0x09, 0xf0, 0x0e, 0xff, 0xfe,
0x1c, 0x14, 0x17, 0x06, 0xe2, 0xf1, 0xfa, 0x01, 0x11, 0x13, 0x12, 0x29,
0xf1, 0x0f, 0x1f, 0xfa, 0xfd, 0xfd, 0x02, 0x07, 0x0e, 0xfb, 0x0e, 0x04,
0x01, 0x01, 0xed, 0xfe, 0xde, 0xfd, 0x08, 0xef, 0xf6, 0x0a, 0xff, 0x0f,

0xe7, 0xf2, 0x0f, 0x02, 0xea, 0x10, 0xf9, 0xec, 0xfd, 0x09, 0xea, 0x1f,
0x46, 0xdd, 0xe2, 0xf7, 0x08, 0xf5, 0xf7, 0xe9, 0x33, 0xfb, 0x2f, 0xf6,
0xb5, 0x1d, 0x15, 0xeb, 0x11, 0xf7, 0x2a, 0x2e, 0x08, 0x1d, 0xf4, 0xfb,
0x15, 0xfa, 0x22, 0x34, 0xff, 0x06, 0xf6, 0xfd, 0xfa, 0xf9, 0x03, 0xf5,
0xf4, 0xf4, 0xd5, 0xea, 0x01, 0x08, 0x22, 0xf1, 0xf2, 0x06, 0xd1, 0xe5,
0x0c, 0xef, 0x12, 0x03, 0x08, 0x02, 0xf7, 0x05, 0x1b, 0x07, 0x39, 0x34,
0x21, 0xe2, 0xe3, 0x0b, 0x0c, 0xf6, 0x29, 0xf7, 0x24, 0x0a, 0xfc, 0xff,
0x1a, 0xfd, 0x05, 0xff, 0xff, 0x0e, 0x0a, 0x1a, 0x09, 0xfb, 0x15, 0x04,
0x03, 0xf7, 0xfe, 0x00, 0xfc, 0xfb, 0x11, 0xfa, 0x1d, 0x0e, 0x06, 0xed,
0xfc, 0x23, 0xd8, 0xf2, 0x04, 0xe5, 0x0f, 0x16, 0x29, 0xfe, 0xf5, 0xec,
0xe2, 0x0e, 0xeb, 0x09, 0x1d, 0x11, 0x05, 0x11, 0xe4, 0x29, 0x12, 0x02,
0x12, 0x19, 0x0e, 0x1a, 0xee, 0xf9, 0x05, 0x09, 0xf5, 0xfd, 0x05, 0x04,
0xe4, 0xf1, 0x17, 0x01, 0xf2, 0xfe, 0x0b, 0xf4, 0x0d, 0x04, 0x06, 0xfe,
0xff, 0xec, 0xe9, 0x00, 0xff, 0x03, 0x03, 0xfd, 0xf1, 0x15, 0xfc, 0xf3,
0xff, 0xfe, 0x09, 0xee, 0x3c, 0x01, 0xec, 0x02, 0xf0, 0xf6, 0x20, 0xeb,
0x16, 0x07, 0x32, 0xf3, 0xce, 0xf0, 0x02, 0xd4, 0x11, 0xe6, 0x28, 0x0e,
0xe3, 0x21, 0xee, 0xce, 0x1e, 0xd9, 0x23, 0x26, 0x06, 0xfa, 0xf9, 0xf1,
0x01, 0xe6, 0x0b, 0x07, 0xdc, 0x21, 0xbc, 0xe3, 0xef, 0xf8, 0x12, 0xfc,
0xe6, 0xfe, 0xf5, 0xd4, 0x15, 0x0a, 0x00, 0x13, 0xfc, 0xec, 0xf3, 0xd6,
0x1a, 0xe3, 0x21, 0x36, 0x2a, 0x03, 0xe9, 0xe3, 0xff, 0x00, 0x13, 0x1c,
0x0e, 0x20, 0xe5, 0xf5, 0x24, 0x0b, 0x20, 0x14, 0x13, 0xf8, 0x04, 0x1b,
0x2f, 0x0a, 0x15, 0x00, 0xf4, 0x1a, 0x11, 0x0d, 0x03, 0x18, 0x0f, 0x18,
0x04, 0x1f, 0xfb, 0xf2, 0x1f, 0x15, 0x03, 0xfb, 0x0b, 0x17, 0xfb, 0x0b,
0x1b, 0x1f, 0xf4, 0x07, 0xf9, 0xf9, 0xf8, 0xf4, 0x14, 0x0f, 0xf6, 0xfe,
0xdd, 0x0b, 0xff, 0x01, 0x18, 0x04, 0x1b, 0x0a, 0xed, 0xe7, 0xf9, 0x16,
0x02, 0x01, 0x00, 0xf7, 0xf1, 0x07, 0xf0, 0x06, 0xf8, 0x0b, 0x02, 0xf3,

0xff, 0x20, 0xfd, 0x01, 0x04, 0xf5, 0xd9, 0xf4, 0xf4, 0xf2, 0xe8, 0xff,
0x04, 0x00, 0xf0, 0xe2, 0xfe, 0xed, 0x1b, 0xef, 0x20, 0xfa, 0xfb, 0xf4,
0x02, 0x18, 0x07, 0xfb, 0xef, 0xe4, 0x08, 0x0d, 0xe1, 0x0e, 0x25, 0xc6,
0xfd, 0x0c, 0x1c, 0x0b, 0xf0, 0x01, 0x1c, 0xd4, 0x11, 0xf5, 0x1b, 0x09,
0xfb, 0xda, 0x13, 0xe3, 0xf9, 0x10, 0x14, 0xf0, 0xf0, 0xfd, 0x1f, 0xcf,
0xf4, 0xe4, 0xfb, 0x0e, 0x0a, 0x11, 0xed, 0xdc, 0xfc, 0xe6, 0xf7, 0xfc,
0x13, 0xe1, 0x0b, 0xe4, 0x04, 0x11, 0xee, 0x21, 0x14, 0xe1, 0x07, 0xe4,
0xfb, 0x08, 0x03, 0x2b, 0x27, 0xf6, 0x0d, 0x02, 0x1b, 0x09, 0x09, 0xf8,
0x14, 0x19, 0x0f, 0x0b, 0x01, 0x10, 0x09, 0x12, 0x03, 0xf5, 0x18, 0xf3,
0xfb, 0xf5, 0x02, 0x0e, 0x0d, 0x00, 0x07, 0xfc, 0x18, 0x25, 0x0b, 0xf0,
0xf9, 0xe6, 0x08, 0x01, 0x24, 0x14, 0xfa, 0xed, 0xe5, 0x1f, 0x09, 0xfe,
0x08, 0xee, 0x1a, 0x1a, 0x05, 0x00, 0xff, 0x0c, 0xfe, 0xf9, 0x11, 0x11,
0xea, 0xfe, 0x08, 0xf9, 0xf0, 0xe4, 0x01, 0x0d, 0xf1, 0x00, 0x0b, 0xea,
0x19, 0xea, 0xf3, 0xf8, 0x08, 0x12, 0x1c, 0x1f, 0xfb, 0xef, 0xf0, 0xf2,
0x14, 0xe1, 0x03, 0xfa, 0xf9, 0xda, 0xe9, 0xfc, 0xf3, 0xff, 0x12, 0x04,
0xf7, 0xfc, 0x17, 0x0f, 0xfc, 0x29, 0x03, 0xe5, 0xf2, 0xee, 0x1e, 0xfa,
0x04, 0xed, 0x25, 0xf4, 0xe1, 0x15, 0x10, 0x1e, 0xef, 0x1c, 0x04, 0xde,
0xe5, 0x08, 0x21, 0xfd, 0xfd, 0xea, 0x03, 0xca, 0xda, 0x26, 0x00, 0x0a,
0xfd, 0x05, 0xf0, 0xd4, 0xe1, 0x1a, 0xe4, 0xf5, 0x07, 0xe7, 0xfa, 0xdf,
0xd4, 0x03, 0xf0, 0x10, 0x15, 0x0c, 0xf4, 0xed, 0xe3, 0xfb, 0x0f, 0x1e,
0x16, 0x09, 0x00, 0xec, 0xea, 0x13, 0x16, 0x0b, 0x01, 0xfb, 0xff, 0x00,
0xfb, 0x07, 0x13, 0x08, 0xf4, 0xe4, 0x12, 0x00, 0xfb, 0xfa, 0xfc, 0x08,
0xeb, 0x19, 0x02, 0x1c, 0xe8, 0x26, 0xf3, 0x10, 0x09, 0x0f, 0x19, 0x02,
0xfb, 0xec, 0xf7, 0xe2, 0xfb, 0xfa, 0x11, 0xf3, 0x0b, 0x08, 0xff, 0xd9,
0xf8, 0x12, 0x18, 0x06, 0x07, 0x22, 0xff, 0x19, 0xf5, 0x0b, 0x0a, 0x13,
0xf2, 0xfa, 0x02, 0x21, 0xeb, 0x11, 0x17, 0x17, 0xec, 0xe1, 0x0e, 0xf7,

0xe8, 0xd8, 0x0e, 0x01, 0xf1, 0xed, 0xed, 0xf0, 0x09, 0xf7, 0xe7, 0xfd,
0xf0, 0xf9, 0xdb, 0xee, 0xdc, 0xfb, 0xf8, 0x0a, 0xf5, 0x0b, 0xd4, 0xd7,
0x08, 0x06, 0x18, 0x06, 0x0c, 0x13, 0xfd, 0x09, 0x13, 0x26, 0x12, 0xf4,
0xef, 0x00, 0xf5, 0x28, 0x18, 0xfe, 0x04, 0x0e, 0x21, 0x1a, 0x0a, 0x1e,
0x09, 0xf0, 0x0d, 0x0f, 0xec, 0xf3, 0x17, 0x22, 0x00, 0xec, 0x0e, 0x01,
0xe9, 0x08, 0x09, 0xf2, 0xf2, 0x08, 0xf0, 0x0b, 0xd9, 0x09, 0x14, 0xf5,
0xf6, 0x04, 0x19, 0xf4, 0x11, 0xe9, 0xf2, 0x0d, 0x20, 0x17, 0x0a, 0x05,
0x0c, 0x04, 0x01, 0xfd, 0xf4, 0xfb, 0x1b, 0x0c, 0xf2, 0x0b, 0xff, 0xfe,
0x01, 0xd8, 0xfa, 0x0e, 0xf5, 0x14, 0xf9, 0x01, 0x04, 0xf8, 0xfa, 0x02,
0xe8, 0xf9, 0xf9, 0xea, 0xf1, 0x07, 0xff, 0x1e, 0x01, 0x0b, 0xf7, 0x0a,
0xf7, 0x0c, 0xfd, 0xec, 0xf3, 0x05, 0xf8, 0xda, 0x0b, 0x15, 0xf6, 0xee,
0xf9, 0x10, 0xfa, 0xfe, 0x08, 0xf0, 0xe6, 0xec, 0x05, 0xff, 0x15, 0x19,
0x1f, 0x11, 0xfc, 0x09, 0x08, 0x01, 0x06, 0xfe, 0x04, 0x08, 0xfb, 0xfb,
0x08, 0xf4, 0xf6, 0x28, 0x10, 0xf9, 0x28, 0x0b, 0xf8, 0x0d, 0x01, 0x00,
0xff, 0x02, 0x05, 0x08, 0xea, 0xe9, 0xf4, 0xf6, 0x01, 0xea, 0xdf, 0x1f,
0xfe, 0x0a, 0xf9, 0xf7, 0x0c, 0x1b, 0x06, 0xed, 0xf6, 0xf2, 0x03, 0x03,
0xfd, 0x04, 0xf5, 0x10, 0x0a, 0x0b, 0xf4, 0xf8, 0xf1, 0xe7, 0x05, 0xfe,
0xe7, 0x0b, 0xf1, 0xec, 0xf4, 0xec, 0x06, 0xee, 0xde, 0x05, 0x1b, 0xfe,
0x13, 0xf3, 0xd9, 0xea, 0x04, 0x10, 0x05, 0xed, 0x15, 0x02, 0x0b, 0x10,
0xfa, 0x02, 0x05, 0x0b, 0x02, 0x07, 0xfc, 0xf5, 0x15, 0x14, 0x05, 0xf7,
0x0c, 0xfe, 0xf6, 0xf4, 0xfa, 0x06, 0xfc, 0x13, 0xdc, 0xe4, 0x09, 0xfa,
0x02, 0x23, 0xec, 0x06, 0x11, 0x13, 0xf8, 0xfa, 0x27, 0x28, 0x0b, 0x23,
0xec, 0xf1, 0x09, 0x17, 0x0f, 0x13, 0xff, 0xf2, 0xfc, 0x0a, 0xf5, 0x0d,
0x03, 0x26, 0x01, 0x0f, 0xfe, 0xf1, 0xfb, 0xe6, 0xf0, 0x02, 0xf2, 0xff,
0x02, 0x11, 0xff, 0xfd, 0x1c, 0x02, 0x0b, 0xf6, 0x14, 0x0c, 0x0b, 0x21,
0x28, 0xf0, 0x11, 0x05, 0x06, 0xed, 0xf9, 0x0a, 0xf2, 0xef, 0xf8, 0xf1,

0xfe, 0x0d, 0xf9, 0xf7, 0xea, 0x00, 0x08, 0xdb, 0x02, 0x0f, 0xfe, 0x04,
0xef, 0x20, 0x16, 0x01, 0xe8, 0xed, 0xe4, 0x22, 0xf6, 0x19, 0x00, 0x04,
0x01, 0x13, 0xeb, 0x0d, 0xec, 0x01, 0x08, 0x05, 0x0c, 0x0e, 0xfe, 0x02,
0x12, 0xf7, 0x27, 0xf9, 0xfd, 0x18, 0xfe, 0x24, 0xf7, 0x13, 0xed, 0x1e,
0x09, 0xff, 0xd8, 0xf4, 0x12, 0xf8, 0x04, 0x0c, 0x1c, 0x11, 0xfd, 0x17,
0x1d, 0x01, 0x13, 0xee, 0x11, 0xf3, 0xf8, 0x06, 0xf6, 0x16, 0xfe, 0x15,
0x16, 0xdc, 0x1f, 0x00, 0x25, 0xee, 0xff, 0xf7, 0xf6, 0x02, 0xdd, 0x15,
0xf1, 0x14, 0x08, 0xe8, 0xe5, 0x21, 0xea, 0xf0, 0x1a, 0x07, 0xea, 0x08,
0xea, 0xe4, 0x1e, 0x00, 0x13, 0x17, 0xec, 0x11, 0xd6, 0x11, 0x18, 0x17,
0x04, 0x15, 0x03, 0x3a, 0xd6, 0x02, 0x07, 0x04, 0xe6, 0xe5, 0xfe, 0x0e,
0xff, 0xed, 0xfc, 0xfb, 0xff, 0x1c, 0x06, 0x0a, 0xfb, 0xf9, 0xea, 0x1a,
0x21, 0xf5, 0x04, 0x06, 0x0a, 0xe3, 0x16, 0xea, 0x04, 0xe2, 0xf9, 0xf9,
0xe6, 0xfb, 0x0f, 0xfc, 0x06, 0xfb, 0x10, 0x07, 0x07, 0x13, 0x07, 0xfc,
0x16, 0xef, 0x07, 0xdc, 0x12, 0x1f, 0x08, 0xf4, 0xe9, 0x14, 0x06, 0xf7,
0xf1, 0x0c, 0x01, 0x0c, 0xe6, 0x04, 0xf3, 0xf2, 0xe5, 0xf3, 0xef, 0x1d,
0xf6, 0x20, 0x07, 0xfe, 0xf4, 0x05, 0xee, 0x10, 0xfd, 0x0e, 0x0b, 0x02,
0x0d, 0xd8, 0x07, 0xfb, 0x26, 0x0a, 0x1c, 0x21, 0x06, 0x1f, 0xf4, 0x06,
0x37, 0x18, 0xfa, 0x16, 0x1e, 0x24, 0xfb, 0xf0, 0x12, 0xf9, 0x02, 0x09,
0x17, 0x16, 0xf3, 0xf9, 0x17, 0xf2, 0x02, 0x0a, 0x2d, 0xe7, 0xe3, 0x25,
0xf0, 0xf9, 0x0f, 0xdd, 0x15, 0xe6, 0x04, 0xfc, 0xf1, 0x17, 0x0a, 0xea,
0x24, 0x07, 0xf1, 0x11, 0x13, 0x29, 0xf4, 0xc5, 0xfb, 0x07, 0xef, 0x13,
0x0b, 0xe1, 0xf1, 0xeb, 0xf8, 0x1b, 0x09, 0x08, 0x1f, 0x15, 0xf2, 0x05,
0x02, 0xdd, 0x09, 0x0f, 0x16, 0x10, 0x01, 0x30, 0xf2, 0xe0, 0x27, 0xfe,
0xf1, 0x0e, 0x0e, 0x07, 0xe6, 0x07, 0x0b, 0x18, 0xfe, 0x0f, 0x01, 0x07,
0xf4, 0x07, 0x10, 0xe7, 0xfb, 0xf3, 0xf7, 0x0b, 0xf9, 0x15, 0x18, 0x25,
0x0c, 0x14, 0x02, 0x08, 0x0a, 0x0f, 0x10, 0xec, 0xee, 0x1a, 0x03, 0x14,

0x0f, 0xfa, 0x25, 0xff, 0x18, 0x0d, 0x0b, 0xea, 0x1f, 0x28, 0x10, 0x0c,
0xe7, 0xee, 0xf7, 0xfa, 0x03, 0x15, 0x0c, 0x1d, 0x01, 0x00, 0x12, 0xee,
0x01, 0xf1, 0xf8, 0x0b, 0xf3, 0xfd, 0x04, 0xf8, 0x02, 0x1e, 0x0e, 0xf3,
0x02, 0x10, 0xfd, 0x07, 0x0b, 0x09, 0x03, 0x10, 0x3e, 0x08, 0x0e, 0x0c,
0xf4, 0xe7, 0xfd, 0x1c, 0x27, 0x1a, 0xed, 0xe1, 0x08, 0xdc, 0xd9, 0xf1,
0x1e, 0x07, 0x12, 0xf1, 0x10, 0xfb, 0xc8, 0x08, 0x0f, 0x03, 0x1d, 0xdc,
0x23, 0x04, 0xf9, 0x0a, 0xff, 0x08, 0x0e, 0xc9, 0x39, 0x0a, 0x01, 0x07,
0xec, 0xe0, 0x05, 0xe8, 0x14, 0xd8, 0xe1, 0xfa, 0xd6, 0xf8, 0xed, 0xdb,
0xff, 0x1d, 0xf5, 0x17, 0x0f, 0x1c, 0xdc, 0xed, 0xff, 0xff, 0x04, 0x13,
0xf5, 0xe7, 0xd2, 0x12, 0xdb, 0xe1, 0x13, 0x11, 0x23, 0x0e, 0xf9, 0x31,
0xdc, 0xef, 0x07, 0x0a, 0x20, 0xf2, 0xf9, 0x13, 0xff, 0x1c, 0x2a, 0xdf,
0xdb, 0xe7, 0x11, 0xf2, 0xfd, 0xfb, 0x28, 0x00, 0x15, 0x03, 0x02, 0x20,
0x07, 0xf7, 0x19, 0x13, 0x13, 0xf6, 0x09, 0xfe, 0xfd, 0x20, 0x14, 0xf5,
0xf5, 0xfc, 0x14, 0x0e, 0x17, 0xfe, 0x15, 0x04, 0xf9, 0xf6, 0x1d, 0xf6,
0x1b, 0xe4, 0xee, 0xfd, 0x00, 0xe9, 0xee, 0xce, 0x0f, 0x20, 0x05, 0x02,
0x0d, 0x06, 0x05, 0xf8, 0xef, 0xdf, 0x16, 0x17, 0xe6, 0xf1, 0x10, 0xf3,
0x06, 0x04, 0xdb, 0xfb, 0xe7, 0xf8, 0x02, 0x11, 0xff, 0x0d, 0x0a, 0xfa,
0x27, 0x0a, 0xfc, 0xe8, 0x11, 0x17, 0xf0, 0x0d, 0x0d, 0xee, 0xdf, 0xdd,
0xf1, 0x15, 0xd6, 0xf7, 0x00, 0xef, 0x2e, 0xe6, 0x24, 0xfd, 0xd5, 0x04,
0xf0, 0x08, 0x08, 0xed, 0x22, 0x07, 0xe1, 0x09, 0xd0, 0x0b, 0x18, 0xe6,
0x3f, 0x0a, 0xe5, 0xe2, 0xf9, 0x08, 0x02, 0xd6, 0x13, 0x15, 0xbd, 0x00,
0x0e, 0xf8, 0xe2, 0xca, 0xec, 0x0e, 0xe6, 0xef, 0x15, 0x11, 0xcb, 0xdf,
0xf9, 0x03, 0x22, 0x10, 0xfb, 0xf9, 0xe5, 0x08, 0xe1, 0x11, 0x10, 0xfc,
0xfa, 0x00, 0xf8, 0x30, 0xe5, 0x08, 0x14, 0xe8, 0x12, 0xe2, 0x04, 0x19,
0x0b, 0xfa, 0x33, 0xf3, 0xec, 0xfe, 0xf8, 0x25, 0xf8, 0x21, 0x28, 0xef,
0x00, 0xde, 0xff, 0x2b, 0x03, 0xfc, 0x10, 0x0c, 0xcf, 0xfd, 0x19, 0x0a,

0x0c, 0xf2, 0xf7, 0x0c, 0xfd, 0x02, 0x1c, 0xdf, 0x26, 0x0d, 0xf0, 0x0b,
0xce, 0x15, 0xfb, 0xec, 0x27, 0xf6, 0xf9, 0xe5, 0xe2, 0xfb, 0xfd, 0xd8,
0x28, 0xec, 0xe9, 0xf2, 0xca, 0x09, 0x02, 0x06, 0x0c, 0xfa, 0x05, 0x01,
0xd5, 0x0a, 0x02, 0xfb, 0x04, 0x17, 0xdd, 0xfe, 0xeb, 0xf1, 0x09, 0x10,
0x12, 0xff, 0x00, 0xe0, 0x26, 0xf7, 0xed, 0xf4, 0x00, 0xf2, 0xfa, 0x07,
0x02, 0xf5, 0x06, 0xe8, 0x03, 0xfd, 0xdc, 0xf2, 0xc2, 0xff, 0x0b, 0xd6,
0x25, 0x04, 0xe9, 0xf0, 0xd9, 0x08, 0x09, 0xc5, 0x23, 0x12, 0xf6, 0x13,
0x11, 0xf3, 0x18, 0xf0, 0x34, 0xfe, 0xfe, 0xed, 0xea, 0x02, 0x17, 0xdc,
0x1b, 0x1b, 0xea, 0xfe, 0xea, 0xfe, 0xf2, 0xc4, 0xfd, 0x04, 0xe9, 0x0d,
0x0d, 0x09, 0xca, 0xd4, 0xe1, 0x04, 0x1e, 0xff, 0x0f, 0xef, 0xd6, 0x0f,
0xd5, 0xf8, 0x26, 0xd6, 0x33, 0xe8, 0xf5, 0x3b, 0xf1, 0xe8, 0x39, 0xe8,
0x08, 0xe5, 0x01, 0x02, 0x04, 0xf6, 0x19, 0x0a, 0xd0, 0xeb, 0x0b, 0x15,
0xf7, 0x0e, 0x23, 0xf6, 0xf4, 0xd8, 0xf4, 0x17, 0x23, 0x25, 0x14, 0x01,
0xd7, 0xfd, 0xf9, 0x1f, 0x1b, 0x11, 0x0a, 0x18, 0xf5, 0xf5, 0x0f, 0xe0,
0x2e, 0x01, 0xe5, 0xdb, 0xe2, 0xf2, 0x14, 0xfa, 0x2a, 0x00, 0xe2, 0xea,
0xfd, 0x0e, 0xfc, 0xc1, 0x35, 0x08, 0xf6, 0xf9, 0xec, 0x00, 0x06, 0x00,
0x0b, 0xf6, 0x01, 0xfe, 0xea, 0x0b, 0x08, 0x05, 0xe4, 0xea, 0xd7, 0xfd,
0xee, 0xf3, 0x0c, 0x0c, 0x0d, 0x02, 0xfd, 0xee, 0x17, 0x10, 0x13, 0xfd,
0x07, 0x03, 0xf8, 0x0c, 0xd4, 0xed, 0xfe, 0x07, 0xf4, 0xee, 0xf4, 0x03,
0xc2, 0x18, 0x2c, 0xd1, 0x33, 0xd8, 0xdb, 0xfa, 0xed, 0x10, 0x1c, 0xe3,
0x37, 0x0a, 0xea, 0xfe, 0xf6, 0xef, 0x20, 0xed, 0x32, 0xf7, 0xf5, 0xf3,
0xca, 0xfd, 0x0a, 0xcf, 0x0d, 0x10, 0xde, 0x07, 0x18, 0x10, 0xf0, 0xd6,
0x0c, 0x04, 0xeb, 0x1a, 0xf9, 0x08, 0xc4, 0xcb, 0xe4, 0x0b, 0x19, 0xfc,
0x29, 0xf6, 0xec, 0x07, 0xf3, 0xed, 0x2b, 0xe9, 0xfa, 0x02, 0xec, 0x2b,
0xf0, 0xf2, 0x2d, 0xe8, 0xed, 0x00, 0x12, 0x13, 0xed, 0x1a, 0x3d, 0xf0,
0x05, 0x04, 0xfc, 0x13, 0x10, 0x01, 0x40, 0xf2, 0x06, 0x02, 0xf9, 0x22,

0x24, 0xff, 0x18, 0x00, 0xeb, 0xe8, 0x14, 0xf9, 0x25, 0xe0, 0xff, 0x03,
0xe5, 0xfd, 0x08, 0xea, 0x2e, 0x0b, 0x05, 0xe7, 0xde, 0xe4, 0xf5, 0xea,
0x3a, 0xf4, 0xf4, 0xe7, 0xed, 0xec, 0xf8, 0xee, 0x30, 0x0a, 0xdb, 0x05,
0xf7, 0x16, 0xff, 0xf7, 0xfa, 0x1f, 0xef, 0xe4, 0xce, 0xf8, 0x13, 0x04,
0xf9, 0x01, 0xe1, 0x03, 0xf9, 0xf9, 0x08, 0x04, 0xfa, 0xe4, 0xe7, 0xf7,
0x28, 0xfd, 0xfd, 0x00, 0xfc, 0xfb, 0xef, 0x0a, 0xec, 0x0c, 0x0a, 0xd2,
0x05, 0xfb, 0xcd, 0xfb, 0x9d, 0xea, 0x1c, 0xe5, 0x25, 0xe8, 0xea, 0x0b,
0xf0, 0xf3, 0x0d, 0xab, 0x49, 0x0e, 0xeb, 0x00, 0xe2, 0x03, 0x29, 0xe0,
0x3d, 0x06, 0xf7, 0xf8, 0xcf, 0x0c, 0x1a, 0xd6, 0x1f, 0xef, 0xfd, 0xff,
0xef, 0x0c, 0xdb, 0xe0, 0x20, 0x06, 0xdf, 0x1a, 0xe7, 0xfc, 0xb2, 0xd1,
0xdf, 0x13, 0x07, 0x1f, 0x0c, 0xf7, 0xde, 0x0a, 0xdb, 0xdf, 0x1a, 0xf5,
0x29, 0x0d, 0xeb, 0x2c, 0xcf, 0x0e, 0x26, 0xfe, 0xef, 0x04, 0xf5, 0x14,
0x09, 0x13, 0x34, 0xff, 0xfe, 0x0e, 0x06, 0x0e, 0x10, 0xf9, 0x2a, 0x0b,
0xe6, 0xfe, 0xf1, 0x1a, 0x36, 0x29, 0x29, 0x05, 0x05, 0xd8, 0x14, 0x12,
0x26, 0x0b, 0x18, 0xff, 0xd7, 0xdf, 0x0f, 0xed, 0x31, 0xf7, 0xfc, 0xec,
0x0b, 0xef, 0x0c, 0xd2, 0x30, 0xf9, 0x04, 0xfe, 0xef, 0xe4, 0xfb, 0xd1,
0x32, 0xe5, 0xee, 0xf0, 0x0c, 0xe6, 0x13, 0xed, 0x1e, 0x0b, 0xe4, 0xe0,
0xfa, 0xf4, 0x14, 0xf4, 0x18, 0xf7, 0xd9, 0xf6, 0xed, 0xea, 0xfc, 0x06,
0xfc, 0xf5, 0xed, 0xeb, 0x05, 0x03, 0x1b, 0x0b, 0xff, 0x0b, 0xef, 0x01,
0xf1, 0x16, 0x05, 0x00, 0xee, 0x0a, 0xdb, 0x10, 0xb4, 0x14, 0x0f, 0xe1,
0x1c, 0xfd, 0xf0, 0xf8, 0xc3, 0x11, 0x17, 0xba, 0x47, 0x15, 0xe6, 0x01,
0xea, 0xf1, 0x0c, 0x08, 0x4a, 0x15, 0xf0, 0xf7, 0xea, 0x00, 0xf5, 0xd4,
0xf1, 0xff, 0xe0, 0x0c, 0xf4, 0x17, 0xd8, 0xea, 0x03, 0xff, 0xd5, 0x18,
0xfb, 0x07, 0xc7, 0xc9, 0xdd, 0xf3, 0x15, 0x0d, 0x22, 0xea, 0xdb, 0x0a,
0xd6, 0x09, 0x1d, 0xe5, 0x2d, 0x04, 0xfc, 0x35, 0xc6, 0x0e, 0x33, 0xf1,
0xd7, 0xea, 0x01, 0x1b, 0x0e, 0x01, 0x2a, 0xff, 0xef, 0xf1, 0xf7, 0x0f,

0xff, 0x00, 0x3b, 0xe8, 0x0a, 0xff, 0xf4, 0x0d, 0x1f, 0x04, 0x17, 0xf7,
0xdf, 0xec, 0x12, 0x26, 0x36, 0x07, 0x0c, 0x06, 0xe7, 0xd6, 0x13, 0xe3,
0x30, 0x09, 0x00, 0xf5, 0xe0, 0xf3, 0x11, 0xe2, 0x38, 0x0d, 0xf6, 0x05,
0xec, 0x05, 0x00, 0xe5, 0x24, 0xef, 0xfe, 0xf8, 0x00, 0xd8, 0x18, 0xf1,
0x26, 0x0b, 0xf2, 0xfc, 0xe0, 0xe4, 0x06, 0x0b, 0x1a, 0x05, 0xc6, 0xf6,
0xe8, 0xde, 0xfe, 0x0c, 0x03, 0x09, 0xfe, 0xe2, 0x18, 0x1b, 0xfb, 0xf7,
0x06, 0xf1, 0xfe, 0xf6, 0xef, 0x1b, 0x07, 0x0d, 0x01, 0x0a, 0xed, 0xf0,
0xad, 0x1a, 0x17, 0xd6, 0x37, 0xfd, 0xd8, 0xec, 0xca, 0xf1, 0x15, 0xc4,
0x33, 0xf1, 0xed, 0xf0, 0xe9, 0x15, 0x0d, 0xf2, 0x36, 0xde, 0xfd, 0x0e,
0xfb, 0x10, 0x0f, 0xf6, 0xf9, 0x0c, 0xea, 0xf0, 0xe5, 0x0b, 0xee, 0xc1,
0x10, 0xf4, 0xe8, 0x1f, 0xee, 0x00, 0xd0, 0xe4, 0xe7, 0x13, 0x07, 0x27,
0x12, 0xea, 0xea, 0x0f, 0xea, 0xf4, 0x14, 0xee, 0xfe, 0x09, 0xfb, 0x31,
0xdb, 0x1b, 0x1c, 0xe7, 0xef, 0xf5, 0xf7, 0x1a, 0x06, 0x01, 0x2c, 0xed,
0xfb, 0x04, 0xfa, 0x07, 0x19, 0xec, 0x2b, 0x0d, 0xfc, 0xd8, 0xfc, 0x0f,
0x1f, 0xfc, 0x2d, 0xf3, 0xc9, 0xda, 0x0a, 0xfe, 0x29, 0x00, 0xfa, 0x09,
0xe8, 0xf6, 0x21, 0xf3, 0x4a, 0x1a, 0xf8, 0x00, 0xe7, 0xf0, 0x21, 0x01,
0x22, 0xf3, 0x00, 0xe9, 0x06, 0xe3, 0x15, 0xd7, 0x3d, 0x0c, 0x07, 0xf1,
0xf3, 0xec, 0x17, 0xdf, 0x29, 0x1b, 0xfd, 0xfe, 0xeb, 0xed, 0x17, 0xf6,
0x23, 0x0a, 0xea, 0xee, 0xf9, 0xf3, 0x0f, 0x0c, 0xf8, 0xf5, 0xed, 0xe8,
0x1c, 0x14, 0x07, 0x17, 0x0b, 0x0d, 0xed, 0xf7, 0xed, 0x10, 0x07, 0xd5,
0xf2, 0x09, 0xd6, 0xf7, 0xb5, 0xf6, 0x19, 0xc9, 0x25, 0x15, 0xe8, 0xf5,
0xc4, 0xf9, 0x2a, 0xb0, 0x39, 0x0e, 0x02, 0x11, 0xf0, 0xf7, 0x1d, 0xeb,
0x39, 0x10, 0x02, 0x15, 0xe0, 0x08, 0x01, 0xee, 0x1c, 0x1e, 0x08, 0x04,
0xf2, 0x02, 0xe8, 0xda, 0xfa, 0xfb, 0xe0, 0xfe, 0x05, 0x02, 0xd3, 0xca,
0xf4, 0xec, 0x10, 0x16, 0x05, 0x0d, 0xd7, 0x09, 0xdc, 0xf6, 0x1e, 0xf8,
0x10, 0xed, 0xf7, 0x27, 0xf5, 0x08, 0x28, 0xee, 0xec, 0xe0, 0xf8, 0x17,

0xfb, 0x23, 0x2e, 0xf1, 0xfa, 0xf5, 0xfc, 0x1a, 0x10, 0xf7, 0x32, 0xfb,
0xfb, 0xe8, 0xf1, 0x03, 0x24, 0xeb, 0x25, 0xf9, 0xca, 0xf1, 0xfe, 0x01,
0x2e, 0x07, 0x18, 0x03, 0xe5, 0xea, 0x10, 0xfa, 0x3b, 0x07, 0x0f, 0x11,
0x04, 0xf7, 0x1d, 0xf1, 0x24, 0xd9, 0x08, 0xef, 0x02, 0xdd, 0x07, 0xc8,
0x2c, 0x0d, 0x06, 0xec, 0x17, 0xda, 0x21, 0xdf, 0x34, 0xd9, 0xfb, 0xf2,
0xf4, 0xec, 0x0e, 0x0a, 0x0f, 0x0f, 0xdb, 0xf0, 0xfb, 0xe6, 0x0f, 0x00,
0x04, 0xf9, 0x01, 0x05, 0x05, 0xfe, 0x08, 0xf3, 0x0e, 0xf2, 0xfb, 0x01,
0xfd, 0x18, 0x1d, 0xf6, 0xee, 0x06, 0xcf, 0xfc, 0xae, 0x27, 0x21, 0xd2,
0x33, 0x03, 0xe0, 0xe0, 0xc9, 0xfb, 0x3a, 0xbd, 0x4d, 0x04, 0xe8, 0xf5,
0xe6, 0xeb, 0x19, 0xf2, 0x4b, 0x1d, 0xfc, 0xf7, 0xd9, 0xff, 0xfe, 0xea,
0x0f, 0x04, 0x0e, 0x00, 0xed, 0x19, 0xe9, 0xe9, 0xff, 0x11, 0xef, 0x14,
0x01, 0x17, 0xbc, 0xb5, 0xef, 0x0c, 0x22, 0x27, 0x0f, 0x01, 0xd4, 0x03,
0xce, 0x01, 0x25, 0xff, 0xf9, 0xf0, 0x0a, 0x1c, 0xe5, 0x0f, 0x1c, 0xee,
0xf4, 0xf1, 0xf4, 0x0c, 0x00, 0x08, 0x1c, 0xf4, 0xd5, 0xf1, 0xfc, 0x1f,
0x11, 0x00, 0x18, 0x03, 0xf7, 0xe4, 0xff, 0x07, 0x09, 0x1a, 0x18, 0xff,
0xea, 0xec, 0xfd, 0x13, 0x2b, 0xf8, 0x0c, 0xfa, 0xdf, 0xf6, 0x11, 0xda,
0x2a, 0xdc, 0xfc, 0xff, 0xff, 0xec, 0x12, 0xe1, 0x37, 0xfd, 0xeb, 0xfe,
0xea, 0xd1, 0x12, 0xfa, 0x28, 0x1a, 0x0d, 0xf0, 0xf7, 0xe0, 0x0c, 0xeb,
0x35, 0x14, 0xeb, 0x00, 0xeb, 0xe7, 0x1b, 0xfc, 0x09, 0x00, 0xf2, 0x04,
0xf9, 0xe5, 0x1a, 0x0e, 0x08, 0x12, 0xf8, 0xfe, 0x09, 0x0f, 0x0d, 0xea,
0x03, 0xe1, 0xfe, 0xf2, 0xec, 0x0d, 0x02, 0xdb, 0x04, 0x1d, 0xd4, 0x01,
0xca, 0x13, 0x29, 0xca, 0x28, 0x04, 0xe2, 0xf1, 0xdb, 0x0b, 0x2c, 0xcd,
0x44, 0x00, 0xe7, 0xf4, 0xd0, 0x12, 0x15, 0xff, 0x42, 0x11, 0x05, 0xfd,
0xd9, 0x11, 0x1c, 0xf4, 0x15, 0xec, 0xf2, 0x24, 0xd6, 0x1d, 0xec, 0xda,
0xf5, 0xec, 0xe5, 0x22, 0xf2, 0x0b, 0xbd, 0xd0, 0xeb, 0x05, 0x07, 0x1b,
0x01, 0xed, 0xf5, 0x02, 0xcf, 0x08, 0x15, 0xfd, 0x1c, 0xe5, 0x04, 0x19,

0xc7, 0x25, 0x22, 0xf3, 0xde, 0xfb, 0xfb, 0x20, 0xf6, 0xeb, 0x25, 0xfe,
0xf5, 0x08, 0xf5, 0x17, 0x0e, 0x04, 0x1c, 0xf9, 0xee, 0xec, 0xe1, 0x06,
0x12, 0xff, 0x2a, 0x13, 0xed, 0xfe, 0x05, 0x18, 0x25, 0x20, 0x09, 0x13,
0xea, 0xd7, 0x05, 0x06, 0x33, 0x25, 0xff, 0x0a, 0xf0, 0xea, 0x17, 0xe1,
0x30, 0xfa, 0x0d, 0x0a, 0x04, 0x00, 0x0e, 0xe9, 0x16, 0x20, 0x0d, 0x02,
0xe8, 0xed, 0x07, 0xe8, 0x3c, 0xf1, 0xd9, 0xfa, 0xe1, 0xed, 0x18, 0xfc,
0xf0, 0x09, 0xe3, 0x05, 0xfe, 0xd1, 0x0b, 0x0e, 0xf5, 0x25, 0xfd, 0xfb,
0x30, 0x1e, 0x08, 0xfc, 0x0c, 0x21, 0xea, 0xfc, 0xe5, 0x1e, 0x16, 0xf5,
0xf4, 0xfc, 0xf0, 0xea, 0xc4, 0x21, 0x27, 0xe9, 0x2b, 0xdb, 0xdb, 0xec,
0xe5, 0xfe, 0x37, 0xe2, 0x46, 0x25, 0xfa, 0xec, 0xe4, 0xf3, 0x19, 0xf2,
0x4c, 0x06, 0x00, 0xfb, 0xeb, 0x10, 0x10, 0xf7, 0x2a, 0xf8, 0xe9, 0x18,
0xee, 0x21, 0xe8, 0xd5, 0xf4, 0x0a, 0xed, 0x24, 0xfe, 0xf9, 0xb2, 0xbc,
0xf3, 0x1d, 0x00, 0x2f, 0x07, 0x08, 0xe1, 0xf1, 0xed, 0x27, 0x27, 0xfe,
0x22, 0xfd, 0x02, 0x20, 0xd8, 0x05, 0x25, 0xec, 0xf1, 0xff, 0x0a, 0x0f,
0xe6, 0xfe, 0x46, 0xfd, 0xe1, 0xca, 0xf7, 0x22, 0x03, 0x08, 0x21, 0xf5,
0x0f, 0xf7, 0xfb, 0x0c, 0xfb, 0x14, 0x2d, 0x03, 0xe5, 0xe4, 0x09, 0x0b,
0x1a, 0xe6, 0x01, 0x28, 0xe9, 0xd6, 0x0b, 0xf7, 0x2c, 0xfb, 0x11, 0xee,
0x0b, 0xed, 0x17, 0xf0, 0x3c, 0xf5, 0x08, 0xfa, 0xf8, 0xcd, 0x17, 0xfa,
0x39, 0xea, 0x11, 0xf5, 0xed, 0xee, 0x0a, 0xec, 0x41, 0xd6, 0xe7, 0xf9,
0xfa, 0xc8, 0x15, 0xf7, 0x08, 0x0e, 0xe3, 0x08, 0xe8, 0xec, 0xfd, 0xfe,
0xf1, 0x00, 0xe9, 0xf4, 0x09, 0x26, 0x02, 0x16, 0xf0, 0x01, 0xef, 0x01,
0xff, 0x03, 0x22, 0xdb, 0xfc, 0xf5, 0xde, 0xe5, 0xc4, 0x01, 0x28, 0xd4,
0x38, 0x08, 0xd0, 0xec, 0xd5, 0x04, 0x2f, 0xce, 0x4e, 0xeb, 0xf9, 0xe7,
0xdf, 0xf0, 0x1b, 0xf5, 0x42, 0xf1, 0xf6, 0x09, 0xd5, 0x0a, 0x0d, 0x08,
0x04, 0x05, 0xe2, 0x0e, 0xd7, 0x19, 0xdb, 0xda, 0xe1, 0x25, 0xde, 0x15,
0x0e, 0x14, 0xbd, 0xb0, 0xe3, 0xe5, 0x24, 0x1e, 0xf8, 0x0d, 0xd8, 0xf7,

0xf2, 0xff, 0x18, 0xf5, 0x07, 0xf0, 0x02, 0x25, 0xd5, 0x1e, 0x2e, 0xdf,
0xe7, 0x05, 0xef, 0x11, 0xe8, 0xe7, 0x47, 0xf4, 0xe1, 0xde, 0x09, 0x36,
0x1a, 0x11, 0x11, 0xf5, 0x12, 0xe5, 0xe7, 0x18, 0x01, 0x17, 0x2a, 0x03,
0x05, 0xea, 0x09, 0x0b, 0x12, 0x04, 0x17, 0xf0, 0xee, 0xd7, 0x11, 0xed,
0x3c, 0x17, 0x16, 0xff, 0x02, 0xdc, 0x21, 0xf3, 0x2e, 0xe5, 0x13, 0xef,
0xec, 0xe2, 0x10, 0xd0, 0x2e, 0xee, 0xff, 0x01, 0xe0, 0xe5, 0x0b, 0xda,
0x1f, 0xf8, 0xf6, 0xfb, 0x07, 0xdb, 0x05, 0xf6, 0x0c, 0xf3, 0xf0, 0x10,
0xf9, 0xf5, 0xf2, 0x0d, 0x10, 0xf7, 0xf6, 0xff, 0x2b, 0x0d, 0x06, 0x1e,
0xf3, 0x0c, 0xe9, 0x01, 0xf2, 0x23, 0xfe, 0xe9, 0xdd, 0x12, 0xdd, 0xf7,
0xbb, 0x22, 0x1b, 0xd4, 0x38, 0x29, 0xd4, 0xcf, 0xf5, 0xf9, 0x27, 0xdd,
0x47, 0x00, 0xf2, 0xe5, 0x09, 0xfc, 0x0e, 0xf9, 0x34, 0x0a, 0x02, 0xfd,
0xec, 0x25, 0x1d, 0x03, 0x15, 0x09, 0xf1, 0x1b, 0xd0, 0x17, 0xda, 0xda,
0xe7, 0x07, 0xe3, 0x15, 0xf1, 0x02, 0xb9, 0xce, 0xe6, 0x0c, 0x10, 0x31,
0xfe, 0xf7, 0xd9, 0xfa, 0xed, 0xed, 0x33, 0xf4, 0x19, 0xe7, 0xfe, 0x3f,
0xe5, 0x06, 0x2e, 0xe6, 0xf2, 0xdc, 0xf5, 0x18, 0xe6, 0x01, 0x2f, 0xee,
0xe7, 0xe4, 0xfe, 0x2c, 0x03, 0xf7, 0x20, 0x05, 0x07, 0xe2, 0x06, 0x1e,
0x05, 0xed, 0x2f, 0x03, 0xea, 0xf8, 0x0e, 0x0c, 0x1f, 0xff, 0x20, 0xf4,
0xe8, 0xe1, 0x1c, 0xec, 0x22, 0x1e, 0x05, 0xfd, 0xf5, 0xca, 0x30, 0xe9,
0x30, 0xe4, 0x14, 0xff, 0xf2, 0xdc, 0x17, 0xf8, 0x26, 0xe1, 0x0b, 0x01,
0x11, 0xc2, 0x02, 0xf1, 0x36, 0x10, 0x02, 0x05, 0xed, 0xf1, 0x15, 0xfa,
0x17, 0xf8, 0xf7, 0xf1, 0xe8, 0xd3, 0xfd, 0x08, 0xfb, 0x27, 0xf5, 0xf5,
0x13, 0x06, 0x0b, 0xf0, 0x01, 0xf9, 0xd7, 0x0e, 0xec, 0x12, 0xfe, 0xfd,
0xee, 0x25, 0xd8, 0xf1, 0xb2, 0x09, 0x1c, 0xbf, 0x34, 0xea, 0xc8, 0xea,
0xdb, 0x0e, 0x24, 0xde, 0x47, 0xfe, 0xdc, 0xe0, 0xf3, 0x06, 0x20, 0xfe,
0x2b, 0xf6, 0x18, 0x14, 0xcd, 0x19, 0x16, 0xfe, 0x1a, 0x15, 0xf8, 0x11,
0xf4, 0x22, 0xd7, 0xcc, 0xdd, 0x15, 0xdc, 0x14, 0xf9, 0x02, 0xbb, 0xca,

0xe3, 0xf3, 0x0d, 0x1e, 0x2a, 0x0c, 0xe4, 0x05, 0xe0, 0x18, 0x2a, 0x07,
0x20, 0xed, 0xf6, 0x17, 0xcf, 0xf4, 0x2a, 0xd6, 0xfb, 0xce, 0x03, 0x37,
0xe2, 0xfd, 0x1d, 0xfb, 0xe5, 0xe0, 0x05, 0x29, 0xef, 0x16, 0x23, 0xf7,
0x01, 0xf4, 0x0c, 0x14, 0xff, 0xee, 0x31, 0xf9, 0x12, 0xf9, 0x14, 0xf6,
0x0c, 0xf6, 0x0b, 0x0f, 0xd8, 0xdc, 0xfe, 0x0f, 0x37, 0xfa, 0x01, 0x09,
0x04, 0xd1, 0x0b, 0x0c, 0x29, 0xf3, 0x0a, 0xf9, 0xed, 0xc2, 0x18, 0xf4,
0x25, 0x18, 0x0f, 0x08, 0xf7, 0xed, 0x1f, 0xf7, 0x4f, 0x0e, 0xf0, 0xe4,
0x00, 0xeb, 0xfa, 0x1a, 0x0c, 0x03, 0xe9, 0xfc, 0xf0, 0xcc, 0x06, 0x05,
0xf2, 0x12, 0x04, 0xe2, 0x16, 0x0a, 0x0a, 0xf3, 0x0b, 0xf3, 0xdc, 0xfd,
0x10, 0xfc, 0x0e, 0xe2, 0xe0, 0xfe, 0xf0, 0xff, 0xb1, 0x06, 0x1b, 0xe4,
0x30, 0x13, 0xc6, 0xc3, 0xfa, 0x0c, 0x1e, 0xd9, 0x57, 0x11, 0xe1, 0xd6,
0xfa, 0xee, 0x1d, 0xf7, 0x37, 0xea, 0xf0, 0x05, 0xef, 0x24, 0x1e, 0xf1,
0x10, 0xe8, 0xeb, 0x19, 0xd1, 0x18, 0xf5, 0xc8, 0xf8, 0xec, 0xf5, 0x1f,
0xf2, 0xff, 0xb3, 0xd2, 0xe6, 0x0e, 0x06, 0x2e, 0x07, 0x17, 0xe0, 0xf5,
0x02, 0xf9, 0x20, 0x07, 0x16, 0x08, 0xe8, 0x1d, 0xd3, 0x08, 0x34, 0xda,
0xf2, 0xce, 0xfb, 0x1f, 0xe1, 0x00, 0x2d, 0xdb, 0xdf, 0xcc, 0x05, 0xfb,
0xf7, 0x00, 0x33, 0xf9, 0x0b, 0x01, 0x13, 0x28, 0xf8, 0x07, 0x24, 0xf8,
0x0f, 0x03, 0x0d, 0xe9, 0x06, 0xfe, 0x18, 0xf9, 0xed, 0xf5, 0x0c, 0xe0,
0x2c, 0x0e, 0xf9, 0x06, 0xfb, 0xce, 0x27, 0xe8, 0x29, 0x19, 0xf9, 0x01,
0x0e, 0xc8, 0x25, 0xed, 0x30, 0xeb, 0x01, 0xfe, 0x10, 0xdc, 0x1e, 0x00,
0x1e, 0x10, 0xf9, 0x00, 0xfc, 0xc8, 0x0e, 0x04, 0x13, 0x04, 0xf0, 0x02,
0xfe, 0xd8, 0x0f, 0x1b, 0xf7, 0xe1, 0xf8, 0xde, 0x12, 0xe2, 0xef, 0x0a,
0x02, 0xe0, 0xdd, 0xf1, 0x0e, 0x2a, 0x25, 0x15, 0xeb, 0x02, 0xf4, 0xf0,
0xbf, 0xfc, 0x27, 0xdc, 0x42, 0x0f, 0xe9, 0xbf, 0xe8, 0x20, 0x33, 0xc9,
0x3f, 0x10, 0xec, 0xf3, 0x03, 0x02, 0x2c, 0x04, 0x38, 0x06, 0x0a, 0xf9,
0xe5, 0x1c, 0x3f, 0x0f, 0x0c, 0x25, 0xe2, 0x06, 0xe6, 0x03, 0xf4, 0xd7,

0xfe, 0xf6, 0xe7, 0x2f, 0xfa, 0x03, 0xb6, 0xcb, 0xf1, 0x11, 0x0a, 0x2c,
0xfc, 0x1e, 0xe0, 0xff, 0xc2, 0xdd, 0x1d, 0xf3, 0x10, 0xfa, 0x07, 0x1e,
0xf6, 0x20, 0x07, 0xe6, 0xf1, 0x0a, 0xe8, 0x27, 0xf1, 0xf5, 0x24, 0xed,
0xfd, 0xee, 0x13, 0x15, 0xe9, 0xe2, 0x22, 0xe5, 0xf9, 0xdd, 0x1d, 0x32,
0x04, 0xfa, 0x25, 0x00, 0xee, 0xfd, 0x0b, 0x0e, 0x23, 0xfa, 0x0f, 0x01,
0xf8, 0xf0, 0x15, 0xe4, 0x21, 0xf7, 0x10, 0xf9, 0xe7, 0xc3, 0x19, 0xe1,
0x34, 0xff, 0xed, 0xf4, 0xef, 0xd7, 0x21, 0x01, 0x31, 0xee, 0xf7, 0xf2,
0xf3, 0xe5, 0x0a, 0xee, 0x2e, 0x1e, 0xf2, 0x0c, 0x07, 0xc2, 0x08, 0x0a,
0x14, 0x14, 0x00, 0xfc, 0xf9, 0xd6, 0xfb, 0xf8, 0xe5, 0xf1, 0xfa, 0xe0,
0x15, 0x21, 0xef, 0x06, 0xf9, 0x00, 0xf5, 0xf4, 0x0b, 0x0b, 0x18, 0x02,
0xf5, 0x04, 0xdb, 0xfd, 0xcc, 0x32, 0x1d, 0xc9, 0x3b, 0x12, 0xd9, 0xaf,
0xcf, 0x0f, 0x26, 0xde, 0x35, 0xe4, 0xdb, 0xd3, 0x22, 0x11, 0x2e, 0xfb,
0x36, 0xfa, 0xfd, 0x02, 0xeb, 0x0f, 0x37, 0x0b, 0x14, 0x1d, 0xdd, 0x18,
0xe0, 0x10, 0xe0, 0xdf, 0x14, 0xf9, 0xf0, 0x19, 0xf7, 0xfb, 0xc4, 0xe5,
0xe7, 0x11, 0x01, 0x31, 0x1a, 0xf7, 0xd8, 0xf1, 0xe9, 0xf3, 0x21, 0xf9,
0xfe, 0xe4, 0xe9, 0x02, 0xd0, 0x06, 0x14, 0xd7, 0xfc, 0xec, 0x06, 0x10,
0xfc, 0xf0, 0x1c, 0xe7, 0xec, 0xe3, 0x03, 0x21, 0xe4, 0x04, 0x12, 0xf0,
0xf3, 0xed, 0x16, 0x36, 0x02, 0xfd, 0x13, 0x11, 0xdf, 0xeb, 0x19, 0x07,
0x10, 0x0c, 0xf9, 0x08, 0xf8, 0xf4, 0x1d, 0xfd, 0x1d, 0x16, 0xf4, 0x0a,
0x08, 0xec, 0x0c, 0x09, 0x3d, 0xe0, 0x0b, 0xee, 0x10, 0xd1, 0x1e, 0x15,
0x43, 0xeb, 0xfa, 0xf3, 0x05, 0xc7, 0xf2, 0xd9, 0x25, 0x20, 0xee, 0xe9,
0xfd, 0xce, 0x16, 0x0c, 0x27, 0x06, 0x0a, 0x06, 0xf9, 0xd6, 0x0b, 0x05,
0xe8, 0x02, 0xe8, 0xd2, 0x10, 0x01, 0xf2, 0x15, 0x09, 0x04, 0xd3, 0xe2,
0xfe, 0xf0, 0x32, 0x1b, 0xd9, 0xf5, 0xea, 0xcc, 0xcb, 0x10, 0x1c, 0xf1,
0x3b, 0x02, 0xd4, 0xbf, 0xca, 0xfe, 0x12, 0xdb, 0x3b, 0xf8, 0xd5, 0xe7,
0x13, 0x10, 0x1a, 0xf4, 0x38, 0x09, 0x08, 0xee, 0xf4, 0xf4, 0x3c, 0xf7,

0x15, 0x04, 0xe4, 0xfa, 0xf4, 0x04, 0xee, 0xf4, 0x07, 0xf8, 0xe9, 0x3b,
0xe2, 0x1f, 0xd5, 0xed, 0xe6, 0xfd, 0x18, 0x49, 0x21, 0x06, 0xd8, 0xde,
0xfa, 0xf0, 0x1b, 0xfe, 0xde, 0x08, 0xf7, 0x14, 0xc7, 0x0f, 0x1d, 0xcf,
0x00, 0xea, 0xff, 0x1b, 0xd5, 0x08, 0x0d, 0xd9, 0xf1, 0xf4, 0x16, 0x23,
0xd8, 0x0c, 0x29, 0xdc, 0xf1, 0xf2, 0x21, 0x49, 0xfc, 0xe2, 0x08, 0x01,
0xf0, 0xf8, 0x17, 0xf9, 0x0f, 0xf5, 0xfa, 0x1a, 0xef, 0xec, 0x09, 0xeb,
0x1a, 0x0c, 0x17, 0x09, 0x11, 0xe9, 0x1a, 0xf7, 0x29, 0xf9, 0xfd, 0x07,
0x01, 0xdd, 0x0a, 0xec, 0x22, 0x15, 0x03, 0xfd, 0xe2, 0xd2, 0x15, 0xec,
0x4d, 0xd7, 0xfc, 0xf6, 0x0b, 0xcc, 0x0e, 0x04, 0x03, 0xf7, 0xfb, 0xfb,
0x0d, 0xeb, 0x19, 0x07, 0xf4, 0xf4, 0xe5, 0xde, 0x22, 0x07, 0xea, 0xf7,
0xeb, 0x23, 0xc8, 0xee, 0x03, 0x04, 0x0f, 0x19, 0xc3, 0xf8, 0x06, 0xd0,
0xf7, 0xfe, 0x0e, 0xe7, 0x0a, 0x02, 0xb0, 0xb8, 0x00, 0xfb, 0x18, 0x0f,
0x22, 0xf7, 0xe9, 0xdc, 0x09, 0x15, 0x23, 0x0d, 0x22, 0x13, 0xe2, 0xed,
0xeb, 0x18, 0x20, 0x0b, 0x12, 0xfc, 0x02, 0xf1, 0xdb, 0x0e, 0xe1, 0x04,
0xdb, 0x0f, 0xf3, 0x1a, 0x06, 0xef, 0xdb, 0xdc, 0xdd, 0xfb, 0x00, 0x2a,
0x20, 0xfd, 0xc1, 0xe3, 0xef, 0x01, 0x14, 0xf2, 0x14, 0x00, 0x0f, 0x28,
0xd9, 0xff, 0xf4, 0xdc, 0x09, 0xfa, 0x1c, 0x08, 0xd1, 0x03, 0x0a, 0xf4,
0xe4, 0xdb, 0x20, 0x30, 0xea, 0x06, 0x11, 0xe2, 0x26, 0xf7, 0x16, 0x22,
0xf9, 0x07, 0x02, 0xf5, 0xf6, 0xfb, 0x1d, 0x0c, 0x16, 0x0a, 0x07, 0xf9,
0x11, 0xde, 0x20, 0x08, 0x19, 0x04, 0x0a, 0x0b, 0x0c, 0xf7, 0xf4, 0xfc,
0x41, 0xf1, 0xf8, 0x16, 0x09, 0xdc, 0x0e, 0x1a, 0x2b, 0x1f, 0xe7, 0xfe,
0x01, 0xe0, 0xfd, 0xe2, 0x34, 0xec, 0xf3, 0xf5, 0x03, 0xec, 0x0b, 0xfb,
0x04, 0xf6, 0xdd, 0xfd, 0x06, 0x14, 0x0d, 0xfa, 0xfc, 0xf1, 0x0a, 0xca,
0x01, 0xec, 0x0e, 0x0e, 0xec, 0xd7, 0xee, 0xd4, 0xf2, 0xfe, 0x16, 0xfa,
0xbd, 0x0d, 0xef, 0xcb, 0xc4, 0xee, 0xed, 0x13, 0x10, 0x19, 0xf8, 0xb1,
0xf1, 0xe3, 0x00, 0xf3, 0x0c, 0xf6, 0xde, 0xc6, 0x15, 0x27, 0x14, 0x29,

0x15, 0xf6, 0xf4, 0xf5, 0xe7, 0x00, 0x0b, 0x2f, 0x0c, 0xef, 0x03, 0x0f,
0xfd, 0x08, 0xf3, 0xf9, 0xf9, 0x05, 0x0d, 0x34, 0x15, 0x1b, 0xc8, 0xd1,
0xf2, 0x1b, 0x0a, 0x22, 0x12, 0x11, 0xe9, 0xf4, 0xe1, 0x2a, 0x20, 0x03,
0xf2, 0xf8, 0x14, 0x0b, 0xd0, 0xf4, 0x0e, 0xbf, 0xc6, 0xd8, 0x04, 0x05,
0xf8, 0xf4, 0x04, 0xc9, 0xea, 0xfd, 0xf7, 0xfa, 0xe3, 0x1b, 0x11, 0xde,
0x0c, 0x11, 0x25, 0x29, 0xe5, 0x02, 0xef, 0xef, 0x02, 0xfa, 0x1a, 0x21,
0x19, 0x09, 0x08, 0x05, 0x04, 0xe5, 0xfa, 0xed, 0x2d, 0x26, 0xfa, 0x17,
0xf6, 0xe8, 0x12, 0x12, 0x31, 0xfc, 0x0d, 0x00, 0xf7, 0xeb, 0x19, 0xf1,
0x2a, 0x06, 0x14, 0xec, 0x08, 0xd3, 0x21, 0x07, 0x32, 0xe3, 0x02, 0x0b,
0xfb, 0xd8, 0x27, 0x07, 0x05, 0xe6, 0xf5, 0xf5, 0x0a, 0xf7, 0x2c, 0x2a,
0xd8, 0x1b, 0xda, 0xf7, 0xea, 0xf6, 0xf9, 0x0e, 0xf8, 0x0c, 0x05, 0xc7,
0xd6, 0x06, 0x12, 0xe3, 0xe1, 0xe1, 0xd8, 0xdb, 0xc6, 0xf8, 0xe6, 0xfa,
0x0c, 0x07, 0xf8, 0xe7, 0xe1, 0x0f, 0x00, 0xf3, 0x03, 0xf0, 0xde, 0xcc,
0xf5, 0xfc, 0xef, 0x1e, 0x16, 0x13, 0xfb, 0xf4, 0x03, 0xe9, 0xfc, 0xfa,
0x15, 0xe8, 0x15, 0x09, 0xf1, 0x0d, 0xdb, 0x0a, 0xe8, 0x09, 0xf5, 0x1a,
0x04, 0xf8, 0xd8, 0xd4, 0x04, 0xee, 0x25, 0x29, 0x09, 0xfe, 0xf3, 0xf5,
0xd4, 0x0a, 0x15, 0x19, 0xf5, 0x12, 0xfe, 0x04, 0xe7, 0x01, 0xeb, 0xde,
0xbe, 0xfe, 0x09, 0x12, 0xdf, 0x13, 0xe0, 0xef, 0xc7, 0xff, 0x03, 0x08,
0xfe, 0xf2, 0x19, 0xe0, 0xe4, 0x0c, 0x22, 0x1e, 0x05, 0xf7, 0x16, 0xf2,
0xf9, 0x06, 0x17, 0xf6, 0x0c, 0x1e, 0x23, 0x08, 0xfe, 0xdc, 0xfd, 0x17,
0x11, 0xdf, 0xf5, 0x0f, 0x01, 0x03, 0x08, 0xee, 0x1b, 0x02, 0x0b, 0x1b,
0x0c, 0x16, 0x1a, 0x00, 0x0f, 0x26, 0x14, 0xf8, 0xf4, 0xf3, 0x19, 0x16,
0x22, 0x0a, 0xd0, 0xf9, 0xf1, 0x05, 0x2b, 0x1e, 0x1e, 0xef, 0xf5, 0x06,
0x05, 0xe7, 0x3f, 0x2a, 0x06, 0xf0, 0x15, 0x14, 0x13, 0x20, 0x1b, 0xde,
0x10, 0x05, 0x33, 0xf8, 0x08, 0x04, 0x17, 0x0d, 0x0f, 0xf6, 0x01, 0xed,
0x28, 0x25, 0x1c, 0x13, 0xfb, 0xea, 0xfb, 0xf3, 0x1c, 0xf9, 0x1f, 0xf0,

0xfb, 0x17, 0xf8, 0xff, 0x10, 0xf7, 0x0b, 0x24, 0x04, 0x00, 0x0d, 0x0c,
0xf7, 0x0a, 0x16, 0x13, 0xf8, 0x05, 0x0a, 0xf1, 0xf5, 0xee, 0xf8, 0x14,
0x0e, 0xed, 0xfe, 0x1b, 0xfe, 0x17, 0x13, 0x10, 0x12, 0x21, 0x1c, 0xfa,
0xe5, 0x0b, 0x08, 0x0c, 0x10, 0x1b, 0x03, 0xef, 0x0d, 0x05, 0x0a, 0xf0,
0x04, 0x11, 0x15, 0x00, 0xfd, 0xef, 0x02, 0x18, 0xf4, 0x09, 0xfa, 0xf6,
0x02, 0xf7, 0xfd, 0x13, 0xef, 0x13, 0xf7, 0xf9, 0x17, 0x0f, 0xfa, 0xf8,
0x15, 0xff, 0x04, 0xef, 0xf0, 0x15, 0xfa, 0xfe, 0xf0, 0xf4, 0xed, 0x06,
0x1c, 0x02, 0xfb, 0xf7, 0x05, 0xfb, 0x0c, 0xef, 0xf4, 0xf0, 0xf6, 0xec,
0x17, 0xf3, 0xf5, 0xef, 0x02, 0xfd, 0xe5, 0x21, 0x0c, 0xf1, 0x1e, 0x08,
0xf1, 0x0b, 0xf7, 0x09, 0x1d, 0xf2, 0xf9, 0xf2, 0xfb, 0x0e, 0xed, 0xf8,
0xfa, 0xdd, 0xf0, 0xfd, 0xdb, 0x1a, 0xf4, 0xef, 0x0c, 0x06, 0x0f, 0xdf,
0xe2, 0x06, 0x06, 0xee, 0xfa, 0x0d, 0x17, 0xfc, 0xf9, 0x15, 0x1a, 0xe4,
0xfb, 0x0c, 0x1a, 0xfc, 0x1b, 0x04, 0x07, 0x20, 0xff, 0x09, 0x0f, 0xf2,
0x26, 0x19, 0x1f, 0x0d, 0x02, 0x16, 0x03, 0x03, 0xfd, 0x05, 0x01, 0x1b,
0x0a, 0x11, 0xfa, 0x21, 0x13, 0xfb, 0x0c, 0x05, 0xf3, 0xdd, 0xe4, 0xdc,
0x22, 0x1b, 0x15, 0x14, 0x0e, 0xe8, 0x00, 0xf7, 0xf8, 0xf4, 0x0b, 0x0b,
0xfd, 0x21, 0xe3, 0x0f, 0xe1, 0x22, 0x01, 0x21, 0x0b, 0x1f, 0x09, 0x10,
0xe2, 0x18, 0x11, 0x0e, 0xed, 0x01, 0x14, 0x12, 0xfd, 0x11, 0xf6, 0xe9,
0x20, 0xe1, 0xf5, 0x1b, 0x27, 0x22, 0xfa, 0xf7, 0xfe, 0x13, 0xf6, 0xdc,
0x06, 0x0d, 0xf4, 0x05, 0x20, 0x0d, 0x0b, 0xe4, 0x15, 0x28, 0x0c, 0x00,
0xf5, 0x07, 0x0c, 0x0a, 0x06, 0x0e, 0xf3, 0xfb, 0xfe, 0x04, 0x08, 0xf4,
0xef, 0x03, 0xe4, 0xeb, 0x06, 0xee, 0xed, 0xdb, 0xeb, 0x1d, 0xf4, 0xfa,
0x0c, 0xfc, 0xfe, 0x11, 0xf7, 0xf8, 0xf5, 0xef, 0xe7, 0xfc, 0x1b, 0xdc,
0x17, 0xfd, 0xfe, 0x00, 0xea, 0xf4, 0xf1, 0xf7, 0x0f, 0x21, 0x04, 0xfd,
0x0d, 0x0c, 0x0a, 0x14, 0xfd, 0x19, 0x09, 0x01, 0xfd, 0xe2, 0x0c, 0x0c,
0xe0, 0x25, 0xfb, 0xff, 0x0d, 0x18, 0xf6, 0x0b, 0x19, 0x12, 0x10, 0x09,

0x0b, 0x06, 0x12, 0x1c, 0x10, 0x03, 0x13, 0x0a, 0x05, 0x0f, 0x09, 0x01,
0x21, 0xe4, 0x01, 0x26, 0xf9, 0xf4, 0x05, 0x19, 0x00, 0xff, 0x0b, 0xff,
0x16, 0x09, 0xe7, 0xee, 0xed, 0xf5, 0x0f, 0x2f, 0xee, 0x19, 0x03, 0x0a,
0x10, 0xee, 0xf7, 0x2e, 0xf4, 0x08, 0xf7, 0xee, 0x07, 0x00, 0xfc, 0x0e,
0xf0, 0x12, 0x08, 0x05, 0xed, 0x11, 0xfc, 0xfb, 0xf7, 0x25, 0xf1, 0x05,
0x0c, 0xf9, 0xfa, 0x03, 0x0c, 0x16, 0x04, 0x25, 0xf8, 0xe7, 0xfc, 0x11,
0x0d, 0x19, 0xd8, 0xfa, 0x0b, 0x06, 0xfd, 0xef, 0x13, 0xf6, 0xff, 0x0e,
0xf9, 0x04, 0xf1, 0xdc, 0xfb, 0xe1, 0xf6, 0x0b, 0x15, 0x07, 0xf7, 0x02,
0x0e, 0xf1, 0xfd, 0xe3, 0xeb, 0x07, 0xf1, 0xef, 0x03, 0xfe, 0xf8, 0x07,
0x10, 0xf7, 0x00, 0xf9, 0xf2, 0x0e, 0xf9, 0xf2, 0x1d, 0xf5, 0xd8, 0xff,
0xe6, 0x18, 0x2a, 0x1b, 0x03, 0x16, 0xfe, 0xf4, 0xf5, 0xfd, 0x04, 0x01,
0xfe, 0xfe, 0x07, 0xfc, 0x0e, 0xfa, 0x15, 0xeb, 0x02, 0x15, 0xea, 0xfd,
0x04, 0xe5, 0xfe, 0xed, 0xfe, 0x1a, 0x09, 0x2a, 0x1b, 0xdf, 0xfb, 0xf8,
0xf1, 0x04, 0x1a, 0x34, 0x07, 0xf9, 0x0d, 0xf5, 0xef, 0xec, 0x10, 0x1a,
0x0b, 0x0f, 0x13, 0xfe, 0x10, 0x22, 0x1e, 0x02, 0xe6, 0xf7, 0x11, 0xfa,
0x11, 0xfc, 0x1b, 0x21, 0x12, 0xf4, 0x18, 0x16, 0x29, 0xe4, 0x0c, 0x2e,
0x12, 0x07, 0x20, 0xf6, 0x1d, 0xf4, 0x12, 0x33, 0xf4, 0xee, 0xfe, 0x05,
0x06, 0xfb, 0x13, 0x0c, 0x0e, 0xf0, 0x00, 0xf8, 0xee, 0xf3, 0x17, 0x00,
0xf7, 0xfb, 0xfc, 0x0f, 0xf4, 0xd5, 0x0a, 0xed, 0xeb, 0xf5, 0xe9, 0xef,
0xd8, 0xf0, 0xf8, 0xe2, 0x19, 0xf7, 0xf8, 0x0a, 0x0b, 0x09, 0xfa, 0xe7,
0x0f, 0xfc, 0xe8, 0x02, 0x00, 0x1a, 0xfe, 0xfd, 0x1b, 0xe6, 0xef, 0x0f,
0xe3, 0x10, 0xf1, 0xe2, 0x0b, 0x0e, 0x06, 0x29, 0x00, 0x01, 0xf3, 0x00,
0x11, 0x04, 0xf2, 0xf7, 0xea, 0xf8, 0xe0, 0x09, 0x0e, 0x13, 0xf4, 0x00,
0x09, 0xfa, 0xf5, 0x0c, 0xff, 0x18, 0x08, 0x0d, 0xfa, 0xde, 0xfa, 0x03,
0xf2, 0xf3, 0x1b, 0xeb, 0x06, 0xea, 0xfb, 0xff, 0x0d, 0xf5, 0x10, 0x17,
0xf8, 0xe8, 0xf1, 0xf1, 0xf5, 0x00, 0x03, 0x0a, 0x09, 0x0a, 0xf3, 0xfb,

0x33, 0x26, 0xe7, 0x17, 0xe3, 0xfa, 0x1f, 0x24, 0xfc, 0x07, 0x02, 0xe2,
0xeb, 0x08, 0x2c, 0xf8, 0x02, 0x1f, 0x04, 0xeb, 0x0b, 0x04, 0x17, 0xf7,
0xff, 0x1c, 0xed, 0x00, 0x3f, 0xd5, 0x17, 0x1d, 0xfe, 0x03, 0xf1, 0x1c,
0x17, 0xec, 0x0e, 0x54, 0xee, 0xf5, 0x25, 0xfa, 0x08, 0xee, 0x13, 0x32,
0x0e, 0xd8, 0x09, 0x0f, 0xee, 0xe5, 0x06, 0x10, 0xf4, 0xfb, 0xe4, 0xfb,
0x09, 0xde, 0x13, 0xff, 0x02, 0xf9, 0xec, 0x0a, 0x00, 0xe9, 0xfd, 0xdc,
0x06, 0x04, 0xdb, 0x06, 0x01, 0xf8, 0x09, 0xe2, 0x0c, 0x14, 0xda, 0xfe,
0x20, 0xe3, 0x09, 0xda, 0x14, 0x12, 0xe1, 0x05, 0xff, 0xf3, 0x00, 0x08,
0xfb, 0xf1, 0xfd, 0xf3, 0x04, 0xfa, 0x08, 0xff, 0x01, 0x1d, 0x0b, 0xfd,
0x0a, 0xf4, 0xfb, 0xfc, 0xf9, 0x19, 0xed, 0xfc, 0xf2, 0x06, 0xe7, 0x02,
0xf6, 0x0c, 0xfc, 0xfb, 0x01, 0x0c, 0xeb, 0x1b, 0xff, 0xff, 0x08, 0x1d,
0xf7, 0xe8, 0xfc, 0xf4, 0x0c, 0xfa, 0xf1, 0xee, 0xed, 0xdd, 0xfc, 0x06,
0x05, 0xdc, 0x1a, 0xfc, 0xf9, 0x07, 0xdf, 0x1b, 0x14, 0x0c, 0xfc, 0x01,
0x16, 0xe1, 0xed, 0x09, 0x34, 0xee, 0xe4, 0x1c, 0x1b, 0xfc, 0x3b, 0x03,
0x15, 0xf2, 0xeb, 0x14, 0x00, 0xdd, 0x24, 0x04, 0xf1, 0xed, 0xfd, 0xe6,
0x32, 0xf9, 0x24, 0x04, 0x0e, 0x22, 0x03, 0x14, 0x2f, 0xf5, 0x1a, 0x37,
0xf4, 0x18, 0x03, 0x0f, 0x4b, 0xe6, 0x0d, 0x5c, 0xf7, 0x1f, 0x1c, 0xe6,
0x23, 0x0c, 0x15, 0x4e, 0xe0, 0x05, 0x1c, 0xec, 0xff, 0x04, 0x13, 0x15,
0xee, 0x07, 0xec, 0x0c, 0xdd, 0xf8, 0x0e, 0x03, 0x0c, 0x1f, 0xe8, 0x0e,
0xf5, 0xec, 0xfc, 0xe2, 0xe8, 0xfb, 0xf6, 0x00, 0xe5, 0xea, 0xf3, 0xd3,
0xf5, 0xfd, 0xd2, 0xfd, 0x1b, 0xed, 0x09, 0xd1, 0x23, 0xfa, 0xd4, 0xf7,
0xe9, 0xf0, 0x0a, 0xd6, 0x14, 0x03, 0xe6, 0x10, 0xf4, 0x18, 0xfe, 0xe1,
0x0b, 0x25, 0xf5, 0xfc, 0xe9, 0xf2, 0xe9, 0xf4, 0x0d, 0xf5, 0x00, 0xf9,
0x17, 0x02, 0xfd, 0x03, 0x04, 0xf8, 0xf5, 0x14, 0xe3, 0xd3, 0xeb, 0xe7,
0x09, 0xf3, 0x14, 0x17, 0xee, 0xe6, 0xf6, 0xff, 0x11, 0x26, 0xf4, 0xf7,
0x02, 0xfa, 0x05, 0x08, 0x16, 0xff, 0x0d, 0xf7, 0xf1, 0xf7, 0xe6, 0xfb,

0x04, 0x04, 0x07, 0x02, 0x04, 0x09, 0xf5, 0xfc, 0x5f, 0xd6, 0xe7, 0x2a,
0x23, 0xf4, 0x1b, 0x06, 0x01, 0xea, 0xe7, 0x05, 0x25, 0xe3, 0x25, 0x07,
0xea, 0xfb, 0xfb, 0x09, 0x25, 0xde, 0x37, 0x04, 0x07, 0xe5, 0xff, 0x14,
0x2f, 0x0a, 0x30, 0x23, 0x04, 0xf0, 0x23, 0xfe, 0x1c, 0xd2, 0x2b, 0x55,
0x01, 0xe5, 0x26, 0xfe, 0x14, 0xed, 0x24, 0x46, 0xe6, 0xee, 0x0f, 0xfd,
0xed, 0xef, 0x0e, 0x1e, 0x05, 0x0a, 0x12, 0xff, 0xe4, 0xf5, 0x0c, 0xed,
0xfd, 0xea, 0x0d, 0x13, 0x1a, 0xe5, 0xfc, 0xc2, 0xef, 0x0a, 0xe2, 0x0f,
0xfe, 0xff, 0x0c, 0xf0, 0xff, 0xdf, 0xea, 0x00, 0xf6, 0xe1, 0x04, 0xd8,
0x26, 0x20, 0xdc, 0xf4, 0x19, 0x06, 0xe8, 0xd2, 0x10, 0x04, 0xf1, 0x02,
0x0c, 0x06, 0xf0, 0xf0, 0x04, 0x1f, 0xf4, 0xf5, 0xed, 0xf1, 0xfa, 0xf1,
0x04, 0x02, 0xf8, 0xfb, 0x04, 0xf1, 0xe5, 0xe4, 0x0a, 0xf0, 0xfe, 0xef,
0x1c, 0xe3, 0xeb, 0xf3, 0x00, 0x17, 0x01, 0x13, 0x19, 0xda, 0xf8, 0x06,
0xde, 0x11, 0xea, 0xf7, 0xf4, 0xef, 0x03, 0x04, 0x0b, 0xe8, 0x08, 0x0e,
0xe2, 0xee, 0xde, 0x06, 0x0e, 0x29, 0xfb, 0xfa, 0x00, 0x02, 0xec, 0x1b,
0x52, 0xff, 0xde, 0x3a, 0x2f, 0x13, 0x30, 0xe9, 0xff, 0xf6, 0xe7, 0x15,
0x1d, 0xd9, 0x3c, 0x0f, 0xe6, 0x14, 0xee, 0x13, 0x1f, 0xe7, 0x33, 0x08,
0xfc, 0x06, 0x0c, 0x08, 0x19, 0xd9, 0x2b, 0x1f, 0x07, 0x10, 0x24, 0x16,
0x29, 0xfc, 0x31, 0x4d, 0xf0, 0xd9, 0x3f, 0xf2, 0x20, 0xe2, 0x25, 0x49,
0xe5, 0xec, 0x0a, 0xf5, 0xf2, 0xd9, 0x22, 0x1f, 0xed, 0x22, 0x02, 0x0a,
0x16, 0x08, 0xf7, 0xfb, 0x0e, 0xfb, 0xfb, 0x1d, 0xf3, 0x1c, 0xf6, 0xe1,
0xcf, 0x19, 0xf4, 0x0f, 0xee, 0xf9, 0x04, 0xd1, 0xf9, 0xe2, 0xda, 0xf1,
0x24, 0xf5, 0x07, 0xdf, 0x1d, 0xf9, 0xdb, 0x18, 0x0b, 0xea, 0x08, 0xca,
0xf2, 0xfa, 0xec, 0x04, 0x0e, 0x17, 0xed, 0xf1, 0x06, 0x15, 0xfc, 0xfd,
0x08, 0xfa, 0xe3, 0xe4, 0x0a, 0xfc, 0xee, 0x08, 0xf5, 0x09, 0xef, 0xee,
0x06, 0xef, 0xe1, 0x19, 0x07, 0xe8, 0xe6, 0xdf, 0xea, 0x0d, 0xf1, 0x16,
0xee, 0xed, 0xf8, 0x09, 0xfa, 0xfb, 0x0c, 0xf8, 0xeb, 0xda, 0x00, 0xfc,

0x04, 0xfe, 0xf5, 0xff, 0xf6, 0xe1, 0x0c, 0x0a, 0x13, 0x0d, 0xf6, 0xf5,
0x15, 0x07, 0xca, 0xec, 0x50, 0x0e, 0xd0, 0x26, 0x4c, 0xf8, 0x23, 0xeb,
0xff, 0x08, 0xe3, 0x11, 0x2c, 0xf9, 0x2a, 0xf1, 0xe9, 0x0b, 0xe9, 0x0f,
0x15, 0xec, 0x33, 0x11, 0x0c, 0x0d, 0x01, 0x01, 0x32, 0xe3, 0x41, 0x27,
0x11, 0x02, 0x2e, 0x07, 0x09, 0xe3, 0x22, 0x4d, 0xf1, 0x05, 0x27, 0x03,
0x25, 0xf5, 0x2c, 0x3b, 0xf4, 0x00, 0x16, 0x0b, 0xec, 0xfe, 0x17, 0x0d,
0xff, 0xe7, 0xfe, 0x24, 0x06, 0xee, 0xf0, 0xe9, 0xfa, 0x1c, 0xf2, 0x19,
0x08, 0xfa, 0xff, 0xd2, 0x01, 0x02, 0xea, 0x05, 0xf2, 0xf4, 0x0b, 0xd2,
0xf9, 0x0d, 0xcd, 0x0d, 0x12, 0xf2, 0x0e, 0xe1, 0x1f, 0x00, 0xe7, 0x14,
0x04, 0xff, 0x09, 0xdb, 0xfc, 0xd9, 0x06, 0xf9, 0xeb, 0x01, 0xef, 0xfa,
0xfb, 0xf5, 0xfc, 0xfb, 0x14, 0xe2, 0xf9, 0xf5, 0x02, 0xfd, 0xfc, 0x01,
0xf7, 0xf3, 0x00, 0xec, 0xe7, 0xf2, 0x00, 0xf1, 0x11, 0xec, 0xf0, 0xe9,
0x11, 0x0a, 0x07, 0x04, 0x01, 0xee, 0xfb, 0xf2, 0x14, 0x01, 0x12, 0xf0,
0xf2, 0xf1, 0xf0, 0xfb, 0x08, 0x03, 0xf8, 0x01, 0xe8, 0xf9, 0x17, 0x26,
0x0f, 0xea, 0xf7, 0xf8, 0x1e, 0xfe, 0xf2, 0xf8, 0x3f, 0x00, 0xd4, 0x1c,
0x53, 0xfe, 0x1e, 0x0f, 0xef, 0xdd, 0xed, 0x10, 0x19, 0xe7, 0x34, 0x0e,
0xde, 0xdf, 0xfa, 0x0e, 0x29, 0xe3, 0x16, 0x09, 0x06, 0x12, 0xeb, 0xf9,
0x32, 0xe0, 0x1a, 0x1d, 0xf3, 0xed, 0x10, 0x07, 0x31, 0xf2, 0x12, 0x52,
0xeb, 0xf7, 0x1e, 0xf7, 0x1a, 0xdc, 0x3e, 0x33, 0xe3, 0xfb, 0x1f, 0x0b,
0x08, 0xfe, 0x13, 0x1a, 0xf4, 0xf8, 0xfe, 0x08, 0xfc, 0xe9, 0xfe, 0xeb,
0xe6, 0xf6, 0x02, 0x18, 0x02, 0xe8, 0xfb, 0xf3, 0x01, 0x08, 0xd7, 0x13,
0x04, 0xe6, 0x02, 0xe6, 0xd7, 0x01, 0xd4, 0xf0, 0x0e, 0x05, 0x18, 0xe5,
0x08, 0xe5, 0xd2, 0x16, 0x12, 0xfe, 0x0e, 0xd3, 0xfc, 0x1f, 0xe9, 0xf8,
0x11, 0x06, 0xf3, 0xd5, 0xf8, 0xff, 0xf0, 0x04, 0x0a, 0xd9, 0xf8, 0xfd,
0xf5, 0x12, 0xff, 0x06, 0x1b, 0xe6, 0xfe, 0xfe, 0xde, 0xee, 0xf6, 0x18,
0xf1, 0xf8, 0x06, 0xf3, 0x02, 0xea, 0x04, 0x14, 0xfc, 0xee, 0xe6, 0x09,

0xf9, 0xee, 0xe3, 0xe7, 0xfc, 0xd9, 0xef, 0xfc, 0x0a, 0x0c, 0x03, 0xf6,
0xe2, 0x11, 0x0f, 0x19, 0x18, 0x10, 0xef, 0xe5, 0x22, 0xf5, 0xe5, 0xe9,
0x4b, 0xf7, 0xdb, 0x0c, 0x4f, 0xde, 0x22, 0x16, 0x09, 0x16, 0xd1, 0xf8,
0x19, 0xe0, 0x24, 0xfe, 0xb8, 0xfb, 0xe5, 0x12, 0x1c, 0xe3, 0x22, 0x09,
0x05, 0x29, 0xf7, 0x10, 0x31, 0xe1, 0x33, 0x3f, 0xfd, 0xed, 0x04, 0x03,
0x2e, 0xed, 0x30, 0x36, 0xee, 0x16, 0x2f, 0xf5, 0x1b, 0xdc, 0x3a, 0x56,
0xe5, 0xef, 0x26, 0xff, 0x03, 0xd7, 0x31, 0x16, 0xef, 0xf1, 0x08, 0x13,
0x01, 0x02, 0x03, 0xf1, 0xf2, 0x08, 0xff, 0x05, 0x12, 0xf2, 0xee, 0xda,
0xed, 0xec, 0xea, 0xf7, 0x0c, 0xf1, 0x09, 0xe6, 0xe6, 0x00, 0xcc, 0x10,
0x0d, 0x0d, 0x20, 0xf4, 0x18, 0x23, 0xec, 0xf9, 0x00, 0xe4, 0x07, 0xd4,
0xfb, 0x16, 0xd2, 0x01, 0xe6, 0x01, 0x06, 0xf0, 0xfe, 0x03, 0xf3, 0x09,
0x01, 0x0d, 0x05, 0xf7, 0xd4, 0x02, 0xfb, 0xfb, 0x08, 0xf0, 0x1f, 0xf3,
0xfe, 0xeb, 0x02, 0x0e, 0x1b, 0x0f, 0x04, 0xf5, 0xf0, 0x1f, 0x14, 0xf7,
0x06, 0xdc, 0xf9, 0xe9, 0x01, 0xff, 0x08, 0xf2, 0x06, 0xff, 0xff, 0xf3,
0x05, 0x1a, 0xfc, 0xfa, 0xeb, 0xfb, 0xfa, 0x12, 0x20, 0xf6, 0xe0, 0xe8,
0x1c, 0xfa, 0xd6, 0x0d, 0x2c, 0x04, 0xe1, 0x09, 0x3b, 0xd3, 0x2a, 0xee,
0xf7, 0xed, 0xf1, 0xf7, 0x0d, 0xf0, 0x32, 0x0f, 0xc9, 0x0e, 0x00, 0x10,
0x24, 0xfb, 0x31, 0xf0, 0xf4, 0xdd, 0xf5, 0x04, 0x25, 0xc7, 0x27, 0x25,
0x16, 0x11, 0x2e, 0x09, 0x30, 0xd1, 0x2c, 0x34, 0xe6, 0xf0, 0x21, 0xf5,
0x21, 0xc8, 0x40, 0x39, 0xde, 0xf0, 0x12, 0xf3, 0x10, 0xe8, 0x1f, 0x18,
0xfa, 0xea, 0x07, 0x11, 0xdf, 0xed, 0xfa, 0xf0, 0x07, 0xef, 0xf3, 0x05,
0x10, 0xe5, 0xf3, 0xe9, 0xe9, 0xe8, 0xd6, 0x01, 0xf9, 0x05, 0x0b, 0xee,
0xf9, 0x12, 0xe3, 0x05, 0xfd, 0xe6, 0x16, 0xe2, 0x1b, 0x12, 0xc5, 0x00,
0xfd, 0x02, 0x04, 0xd2, 0xff, 0xec, 0xf6, 0xfd, 0x00, 0xe4, 0xf7, 0xf3,
0xeb, 0xfa, 0xf8, 0x0d, 0x03, 0xfa, 0xfe, 0xe4, 0xdb, 0xe3, 0x06, 0xff,
0xf4, 0xf2, 0x1b, 0xf1, 0xf7, 0x02, 0x01, 0x04, 0x13, 0xe5, 0x0c, 0x05,

0xf7, 0x0a, 0x03, 0x03, 0x0b, 0x03, 0xee, 0xf7, 0x21, 0x20, 0xff, 0xf3,
0x09, 0xe5, 0xff, 0xec, 0x17, 0x00, 0x06, 0x14, 0xeb, 0xf2, 0x18, 0x16,
0x1f, 0xec, 0xee, 0xe1, 0x1e, 0x03, 0xfa, 0xfe, 0x28, 0x03, 0xc9, 0x0c,
0x3f, 0xd8, 0x30, 0x16, 0x03, 0xf8, 0xe9, 0xfb, 0x28, 0xe1, 0x36, 0x0a,
0xdf, 0xe5, 0xeb, 0x08, 0x1c, 0xcd, 0x29, 0xf2, 0xfc, 0x0a, 0xed, 0x01,
0x29, 0xf1, 0x20, 0x13, 0x04, 0xec, 0x17, 0x0a, 0x35, 0xc3, 0x1a, 0x46,
0xe0, 0xd7, 0x3c, 0x09, 0x28, 0xd1, 0x22, 0x20, 0xd5, 0xfa, 0x28, 0xfa,
0xff, 0xea, 0x1d, 0x23, 0xe0, 0x07, 0x07, 0x0f, 0xf1, 0xf1, 0x08, 0xf0,
0xf8, 0xff, 0x05, 0x1b, 0x05, 0xfa, 0xf0, 0xfb, 0xe3, 0xe4, 0xcc, 0x1a,
0xf9, 0x09, 0x06, 0xee, 0xf4, 0x03, 0xd0, 0x14, 0xf4, 0xff, 0x1d, 0xe8,
0x11, 0xf4, 0xd1, 0xf4, 0x04, 0x0b, 0xfb, 0xdc, 0x0a, 0x0c, 0xeb, 0xed,
0x06, 0xf3, 0x04, 0xdd, 0xdf, 0xf9, 0xea, 0xfc, 0xf5, 0xf2, 0xfb, 0xea,
0xe3, 0x03, 0xee, 0x0e, 0xff, 0xdb, 0x1e, 0x04, 0xf7, 0x1a, 0x04, 0x0c,
0x0d, 0xda, 0x04, 0xe9, 0xff, 0x04, 0x00, 0x0c, 0xf9, 0xe4, 0xfb, 0xf6,
0x14, 0xde, 0x1b, 0x00, 0x0b, 0xfe, 0x06, 0xf8, 0x0f, 0xdc, 0x01, 0xef,
0xef, 0x0d, 0xf8, 0xf1, 0x0f, 0xf9, 0xf9, 0xdf, 0x0d, 0xe4, 0xd9, 0xf9,
0x2b, 0xee, 0xe8, 0x09, 0x40, 0xf9, 0x2f, 0x0a, 0xfa, 0xe8, 0xe9, 0x01,
0x0e, 0xe7, 0x23, 0x0a, 0xd0, 0x19, 0xd3, 0x0e, 0x04, 0xda, 0x2b, 0x0f,
0xe7, 0xe6, 0xf3, 0xfb, 0x2c, 0xd3, 0x36, 0x19, 0x0e, 0xfe, 0x03, 0x1a,
0x2e, 0xd0, 0x23, 0x32, 0xf1, 0xe1, 0x2a, 0x09, 0x1b, 0xf6, 0x29, 0x3e,
0xce, 0x15, 0x0a, 0xe8, 0xec, 0xdf, 0x44, 0x28, 0xd9, 0xfd, 0xfa, 0x09,
0xff, 0xe7, 0x08, 0xec, 0xf4, 0xef, 0x01, 0x19, 0x11, 0xf3, 0xeb, 0xeb,
0xed, 0x1a, 0xdd, 0x15, 0x0f, 0x07, 0xfe, 0xeb, 0xff, 0xd6, 0xd5, 0x04,
0xf5, 0x07, 0x10, 0xe6, 0x0c, 0xe4, 0xda, 0x0c, 0x08, 0xee, 0x06, 0xd8,
0xf8, 0xf1, 0xe0, 0x01, 0x08, 0xfe, 0xf9, 0xf3, 0xdf, 0x03, 0xe6, 0xf4,
0x0a, 0xff, 0xf2, 0xe0, 0xd9, 0xeb, 0x01, 0x10, 0x02, 0xfc, 0x0d, 0x14,

0xea, 0xf8, 0x03, 0x18, 0xf3, 0x09, 0xfc, 0x0c, 0x0b, 0x1f, 0xf5, 0x05,
0xf7, 0xf9, 0x00, 0xfd, 0x04, 0xfc, 0x16, 0x07, 0x00, 0xdf, 0xf9, 0xfa,
0x0c, 0xfb, 0xf4, 0xf7, 0xf0, 0xeb, 0x07, 0x17, 0x20, 0xfb, 0xf0, 0xec,
0x04, 0x00, 0xf8, 0xf2, 0x2d, 0xf9, 0xd9, 0x0b, 0x55, 0xec, 0x33, 0x26,
0xf8, 0x0a, 0xf2, 0x0b, 0x25, 0xdf, 0x29, 0x05, 0xd1, 0x14, 0xe2, 0xf2,
0x12, 0xdd, 0x28, 0xfc, 0xec, 0x08, 0xfd, 0x02, 0x3a, 0xe6, 0x29, 0x25,
0x0d, 0x10, 0x09, 0x0a, 0x32, 0xf5, 0x17, 0x2d, 0xea, 0xfb, 0x35, 0xfc,
0x28, 0xd0, 0x29, 0x2f, 0xcb, 0x06, 0x0f, 0x04, 0xf2, 0xf3, 0x34, 0x1c,
0xf4, 0x08, 0x05, 0xfc, 0xfd, 0xed, 0x0f, 0xf8, 0xe9, 0xf0, 0x09, 0x16,
0xfe, 0x02, 0xff, 0xd4, 0xea, 0x0a, 0xeb, 0x0c, 0xf8, 0xf4, 0x09, 0xf4,
0xf2, 0x07, 0xd9, 0x0b, 0xfd, 0xe4, 0x1a, 0xef, 0x14, 0x08, 0xd8, 0xfc,
0xf5, 0xe1, 0x03, 0xcf, 0xf1, 0x11, 0xdb, 0x15, 0x07, 0x10, 0xf8, 0xfc,
0xe2, 0xf1, 0xf5, 0xde, 0xff, 0xe7, 0x01, 0xea, 0xee, 0xe9, 0x02, 0x0a,
0x18, 0xec, 0xfe, 0xf9, 0x09, 0xf3, 0x0e, 0x02, 0xf1, 0xfc, 0xf9, 0x16,
0x05, 0x07, 0x09, 0x0d, 0x0e, 0xf7, 0x04, 0xed, 0x04, 0xdb, 0x04, 0x04,
0xf6, 0xdc, 0xee, 0xec, 0xf5, 0xfe, 0xf4, 0x02, 0xe4, 0x0b, 0xe0, 0x17,
0x0a, 0xe0, 0xf7, 0xdc, 0x11, 0xd6, 0xfe, 0xfa, 0x35, 0xde, 0xe6, 0x06,
0x44, 0xf9, 0x35, 0x0a, 0xfb, 0xff, 0xec, 0xfb, 0x16, 0xd9, 0x23, 0x0f,
0xd4, 0xef, 0xdf, 0x06, 0x0b, 0xd9, 0x25, 0xff, 0xf8, 0xeb, 0xf4, 0x0a,
0x20, 0xe5, 0x22, 0x1c, 0xeb, 0xf4, 0x0d, 0x0c, 0x19, 0xe1, 0x1e, 0x31,
0xe9, 0xfb, 0x20, 0xf0, 0x23, 0xfe, 0x35, 0x28, 0xb4, 0x06, 0x28, 0xe7,
0xfb, 0xe9, 0x2a, 0x1a, 0xef, 0x15, 0x0c, 0xed, 0xf1, 0x04, 0x0e, 0x0a,
0xff, 0x16, 0x01, 0x04, 0x17, 0xea, 0xec, 0xdc, 0xf4, 0xf7, 0x04, 0x16,
0x1f, 0x0a, 0x11, 0xef, 0x12, 0xdf, 0xd9, 0x0c, 0xf5, 0x10, 0x02, 0xf3,
0x10, 0x03, 0xd3, 0xf5, 0x0b, 0x02, 0x00, 0xcb, 0xf6, 0x23, 0xf6, 0xf1,
0x1f, 0xf9, 0xfc, 0xf0, 0xf6, 0xfe, 0xfa, 0xf8, 0xf9, 0xf4, 0xfb, 0x0a,

0xd6, 0x29, 0x09, 0x02, 0x00, 0xfc, 0xfc, 0xee, 0xf5, 0x05, 0xfb, 0x1e,
0xf1, 0xf1, 0xf3, 0x02, 0xec, 0x1c, 0x0c, 0x0e, 0x0b, 0x04, 0xf6, 0xe7,
0x14, 0x08, 0x27, 0x01, 0xfe, 0xe5, 0xe7, 0x01, 0x1b, 0xf0, 0xf6, 0xff,
0xf4, 0xe7, 0xee, 0x18, 0x0d, 0x08, 0xf8, 0xd6, 0x07, 0xf4, 0x08, 0xff,
0x1d, 0x13, 0xe7, 0x0b, 0x42, 0xef, 0x28, 0x00, 0xf9, 0xf0, 0xf3, 0x00,
0x15, 0xfd, 0x1a, 0x22, 0xc1, 0xf5, 0xe0, 0xf8, 0x09, 0xe6, 0x0e, 0x05,
0xf9, 0xf6, 0x01, 0x01, 0x13, 0xdc, 0x1f, 0x0d, 0xfb, 0x04, 0x08, 0x0b,
0x15, 0xdb, 0x28, 0x34, 0xed, 0x0b, 0x3a, 0xed, 0x16, 0xe3, 0x39, 0x32,
0xc4, 0x0b, 0x20, 0xe7, 0xf7, 0x02, 0x35, 0x24, 0xfc, 0xe8, 0x1c, 0xf8,
0xf1, 0xfa, 0x0c, 0x1d, 0xf2, 0x05, 0xff, 0x12, 0x0f, 0x01, 0xec, 0xea,
0xf0, 0x03, 0xe7, 0x15, 0xfd, 0x05, 0x08, 0xe0, 0x1b, 0xf8, 0xe1, 0x1e,
0xed, 0xdc, 0x11, 0xeb, 0xfd, 0x1a, 0xeb, 0x09, 0xf9, 0xf3, 0x00, 0xe8,
0xe6, 0x08, 0xf7, 0xde, 0x1e, 0x00, 0x00, 0x00, 0xe4, 0x09, 0xf2, 0xf8,
0xe7, 0xf2, 0x0d, 0xfa, 0xe2, 0x0f, 0x04, 0x08, 0xf2, 0x13, 0xf8, 0xf9,
0xf1, 0xff, 0x03, 0x11, 0x12, 0xe9, 0xf4, 0x13, 0x07, 0x0c, 0x13, 0x2b,
0xf7, 0xdd, 0xf9, 0xe9, 0xfa, 0xdb, 0x1d, 0xf6, 0xf6, 0xf9, 0xe4, 0xf6,
0x0d, 0xeb, 0x0d, 0x08, 0xe7, 0xe7, 0xf2, 0x03, 0x1d, 0xd9, 0xd8, 0xe4,
0xf7, 0xea, 0xdc, 0xdc, 0x26, 0x02, 0xee, 0xfa, 0x38, 0xfc, 0x1a, 0xef,
0xda, 0xf1, 0xdf, 0x0b, 0x1a, 0xe0, 0x16, 0x16, 0xdc, 0x04, 0xfa, 0xf7,
0xee, 0x02, 0x25, 0x02, 0xf5, 0xfb, 0x08, 0xf6, 0x11, 0xf5, 0x12, 0x08,
0xf4, 0xe3, 0x1b, 0xf5, 0x3a, 0xdc, 0x20, 0x2e, 0xe0, 0xf5, 0x30, 0xe4,
0x09, 0xf8, 0x3c, 0x45, 0xd3, 0x08, 0x23, 0xd8, 0x09, 0xe4, 0x35, 0x30,
0xe4, 0xfe, 0x07, 0xf6, 0x05, 0x01, 0x05, 0xff, 0xf6, 0x0d, 0x02, 0xfd,
0x03, 0x05, 0x0d, 0x00, 0xf5, 0xd6, 0xcf, 0x19, 0x06, 0xee, 0x0d, 0xf2,
0x01, 0x18, 0xef, 0x12, 0x04, 0x02, 0x21, 0xd9, 0x02, 0x0d, 0xeb, 0xe9,
0x13, 0x08, 0x15, 0xf0, 0xee, 0x03, 0xec, 0x06, 0x17, 0xed, 0x00, 0x1a,

0xee, 0xf2, 0xfc, 0x09, 0xec, 0xf8, 0xf8, 0x18, 0xf4, 0x13, 0x04, 0xf6,
0x02, 0xf0, 0xfc, 0xfe, 0xe3, 0x01, 0x0a, 0x1c, 0x1b, 0xec, 0x0e, 0x01,
0xfb, 0x08, 0x11, 0xf5, 0x00, 0x14, 0xe6, 0x12, 0x07, 0xf4, 0x15, 0x07,
0xfc, 0xfb, 0xf5, 0xf1, 0x01, 0x21, 0x01, 0xe9, 0xe8, 0xef, 0xdb, 0xdf,
0x1f, 0x0a, 0xdd, 0xd1, 0x16, 0x04, 0xfd, 0xe1, 0x24, 0xf0, 0xec, 0xf4,
0x38, 0xe1, 0x16, 0xfd, 0xe0, 0xec, 0xe7, 0x0c, 0x2a, 0x04, 0x0c, 0x17,
0xdc, 0xe8, 0xf2, 0x03, 0xec, 0xfd, 0x19, 0xfe, 0xf3, 0xf0, 0xf3, 0xfb,
0x18, 0xdf, 0x1c, 0x00, 0x09, 0xf4, 0x18, 0x0b, 0x1f, 0xf6, 0x34, 0x22,
0xf4, 0x22, 0x45, 0xeb, 0x23, 0xcf, 0x32, 0x34, 0xf2, 0xf9, 0x29, 0xd4,
0xf7, 0x0b, 0x38, 0x2a, 0x09, 0xe6, 0x05, 0x01, 0x0b, 0xfe, 0x17, 0xfb,
0x00, 0xeb, 0x08, 0xfd, 0x0c, 0x02, 0x1d, 0xea, 0xfa, 0x0b, 0xeb, 0x09,
0xfe, 0xfe, 0x10, 0xe0, 0xf6, 0x06, 0xf0, 0x15, 0xf3, 0x09, 0x11, 0xe4,
0xf9, 0x07, 0xe1, 0xed, 0x17, 0x05, 0x0c, 0xe1, 0xdb, 0xf2, 0xf8, 0xea,
0x22, 0xe9, 0x02, 0x00, 0xfd, 0xe7, 0xf2, 0xf8, 0xf9, 0xfc, 0xfa, 0xe8,
0xe8, 0xeb, 0xe9, 0x0d, 0x04, 0xf8, 0xf8, 0xf7, 0xf8, 0x0d, 0x03, 0x0c,
0x13, 0xf2, 0x0f, 0xf9, 0xe6, 0xfd, 0x0f, 0x19, 0x08, 0xf7, 0xfa, 0x01,
0xf3, 0x12, 0x1e, 0x05, 0x0a, 0x09, 0xfd, 0x0b, 0x07, 0x08, 0x02, 0xfc,
0xd6, 0xe8, 0x14, 0x01, 0x13, 0x19, 0xef, 0xda, 0x0e, 0x0a, 0x07, 0xef,
0x34, 0xe0, 0x05, 0x1e, 0x4e, 0xe9, 0x19, 0xff, 0xe1, 0x04, 0xfb, 0x0e,
0x11, 0x05, 0x1f, 0x15, 0xd4, 0xec, 0xf9, 0xe7, 0xf9, 0xfc, 0x25, 0xff,
0x06, 0xf2, 0x01, 0xf6, 0x2a, 0x17, 0x24, 0x11, 0xf3, 0x1a, 0x1f, 0xfb,
0x32, 0xeb, 0x33, 0x2f, 0x00, 0x08, 0x2c, 0xf0, 0x26, 0xf4, 0x25, 0x36,
0xd9, 0xf1, 0x1a, 0xd5, 0xec, 0xf9, 0x32, 0x27, 0xfc, 0xf4, 0xf0, 0xe3,
0xfa, 0x0c, 0x16, 0x17, 0xfa, 0xf9, 0xe5, 0x1f, 0x1f, 0xfa, 0xff, 0xfd,
0x0d, 0x02, 0xe9, 0x0e, 0xf0, 0x12, 0x09, 0xda, 0x02, 0xea, 0xe5, 0x0a,
0xff, 0x03, 0x13, 0xf0, 0x0a, 0xf9, 0xe9, 0xff, 0x10, 0xfc, 0x1a, 0xf3,

0xf7, 0x0f, 0xf4, 0xfa, 0xf4, 0x05, 0x10, 0x0a, 0xdd, 0x09, 0xf7, 0xf0,
0xe5, 0x07, 0x07, 0xfa, 0x02, 0xd7, 0xf8, 0xf7, 0x01, 0xfb, 0x0e, 0xf8,
0x07, 0x0f, 0xfe, 0x03, 0x12, 0x05, 0x09, 0x13, 0xf8, 0xdc, 0xfd, 0x27,
0x0f, 0xec, 0xf7, 0x07, 0x00, 0xfc, 0x12, 0xf8, 0xfb, 0xea, 0xe4, 0xe9,
0xe9, 0xe0, 0xff, 0xdc, 0xd6, 0xeb, 0xf2, 0xf7, 0x0d, 0x1b, 0xe9, 0xc4,
0x06, 0x00, 0xfd, 0x04, 0x46, 0xf9, 0xe9, 0x13, 0x2d, 0x0c, 0x1f, 0xf8,
0xd3, 0x0c, 0x14, 0x11, 0x05, 0xe5, 0x27, 0x08, 0xc5, 0xef, 0xdf, 0xdd,
0x04, 0xf8, 0x11, 0x10, 0xf0, 0xe7, 0xfb, 0x03, 0x3c, 0xe7, 0x14, 0x0c,
0xf4, 0xf6, 0x1b, 0x0a, 0x23, 0xf2, 0x2d, 0x1a, 0x08, 0xff, 0x32, 0xe7,
0x1a, 0x05, 0x2b, 0x34, 0xf1, 0x0a, 0x00, 0xe8, 0x02, 0xdf, 0x2c, 0x2a,
0x03, 0xe6, 0xfc, 0xef, 0xfc, 0xe4, 0x03, 0x01, 0x03, 0xee, 0xe9, 0x15,
0x05, 0x03, 0x13, 0x11, 0x0e, 0xee, 0xf5, 0x22, 0x1b, 0x0e, 0xfd, 0xf3,
0x0a, 0x02, 0xdd, 0x20, 0xeb, 0x06, 0xf8, 0xe2, 0x06, 0x0e, 0xde, 0x0d,
0xf9, 0x16, 0x1c, 0x0c, 0xe0, 0xf0, 0xec, 0x0c, 0x0f, 0xf2, 0x27, 0x1d,
0xde, 0xe6, 0xf0, 0xf9, 0xf0, 0x02, 0x0a, 0x07, 0x06, 0xf9, 0x0f, 0xfa,
0xf0, 0xee, 0xf1, 0xf7, 0xff, 0x02, 0x0b, 0x0d, 0x1b, 0xee, 0xf6, 0x05,
0xff, 0x1c, 0x17, 0x04, 0x05, 0x17, 0x00, 0xff, 0x0d, 0xf3, 0x23, 0x10,
0xfd, 0x05, 0xfb, 0xea, 0x03, 0x10, 0x07, 0xd7, 0xf7, 0xff, 0xf3, 0xf1,
0x17, 0xed, 0xd3, 0xcb, 0x14, 0x1c, 0xf5, 0x03, 0x47, 0xf6, 0xf7, 0xf2,
0x3e, 0xf2, 0x22, 0xf4, 0xed, 0xfc, 0xee, 0x0b, 0xf4, 0xf1, 0x25, 0x10,
0xd0, 0xf6, 0x00, 0xef, 0x10, 0xfc, 0x15, 0xe5, 0xdb, 0xf3, 0xea, 0x10,
0x22, 0xf2, 0x2b, 0x11, 0xf9, 0x0a, 0xfc, 0xf5, 0x53, 0x16, 0x25, 0x43,
0xe0, 0x0e, 0x13, 0xfc, 0x2d, 0xe2, 0x55, 0x65, 0xf4, 0x08, 0x01, 0xdf,
0x0a, 0x00, 0x49, 0x1c, 0xfe, 0xdf, 0xef, 0xf2, 0xf9, 0xf6, 0xfd, 0xff,
0xf3, 0x02, 0xf6, 0x14, 0x0b, 0xe8, 0x09, 0xfc, 0xfc, 0xe2, 0xe5, 0x11,
0x03, 0x09, 0xfb, 0x06, 0x10, 0x1a, 0xf3, 0x0d, 0xfa, 0x0a, 0xd5, 0xf5,

0x1a, 0x11, 0xf2, 0xfc, 0x1f, 0xfe, 0x0e, 0xe4, 0xef, 0xd7, 0xee, 0x06,
0x1e, 0x04, 0x12, 0x28, 0xf7, 0x0e, 0x06, 0xf8, 0xee, 0xf0, 0x1a, 0x01,
0xf7, 0xfd, 0x03, 0x11, 0x19, 0x10, 0x04, 0xfb, 0xd7, 0xfa, 0x16, 0x06,
0x07, 0x23, 0xfa, 0x14, 0x11, 0xf1, 0x12, 0x10, 0x04, 0xe1, 0xee, 0xf7,
0x21, 0x0e, 0x0a, 0x0a, 0xf8, 0x07, 0x0a, 0xee, 0x03, 0x1f, 0xfa, 0xc4,
0xec, 0x12, 0x01, 0x1e, 0xfd, 0xf1, 0xe8, 0xcc, 0xf4, 0x17, 0xff, 0xdd,
0x45, 0x10, 0xee, 0xfa, 0x3d, 0xe7, 0x27, 0xdd, 0xd7, 0xf9, 0xf4, 0xf6,
0x06, 0xf8, 0x1e, 0x13, 0xe7, 0xe2, 0xf1, 0xe3, 0xf3, 0xf7, 0x18, 0x12,
0xe4, 0x0a, 0xdb, 0xff, 0xff, 0xfe, 0x20, 0x09, 0x00, 0xf7, 0x23, 0xf6,
0x2d, 0x14, 0x26, 0x28, 0xe5, 0xff, 0x0f, 0xe3, 0x1d, 0xe8, 0x56, 0x43,
0xe7, 0xfb, 0xf9, 0xe6, 0xe9, 0xe2, 0x19, 0x19, 0x08, 0xfa, 0xf3, 0xe5,
0x23, 0x07, 0x0f, 0xf8, 0xf8, 0xf3, 0xfc, 0x11, 0x2a, 0x05, 0xf4, 0xf1,
0xfa, 0xfb, 0xf1, 0x1e, 0x13, 0x0f, 0xf9, 0xf5, 0xfa, 0x09, 0xf9, 0x03,
0xf0, 0xf0, 0xe7, 0xec, 0xf1, 0x0c, 0xe6, 0xee, 0xf6, 0x20, 0x0f, 0xe9,
0x00, 0xf4, 0xfe, 0xf0, 0x13, 0x0a, 0x17, 0x13, 0xee, 0x13, 0xfb, 0xff,
0xf8, 0xfd, 0xf4, 0xe2, 0xe8, 0x06, 0xfc, 0x14, 0x03, 0x17, 0x00, 0x03,
0xe6, 0xfd, 0xf2, 0x12, 0x12, 0x20, 0xeb, 0x10, 0x02, 0xf7, 0x13, 0x0d,
0x11, 0xfd, 0xde, 0xf5, 0x07, 0xf3, 0x04, 0xff, 0x06, 0x05, 0xfb, 0xea,
0xf0, 0x0a, 0x00, 0xb5, 0xe8, 0x1a, 0x03, 0xfe, 0x0d, 0x1a, 0xe7, 0xc0,
0xd6, 0xdc, 0xf6, 0xf8, 0x39, 0xf5, 0xd5, 0xf8, 0x22, 0xfa, 0x22, 0x05,
0xd0, 0xf4, 0x2d, 0xfc, 0x00, 0x0a, 0x1b, 0xfc, 0xe6, 0x09, 0x14, 0xfa,
0x00, 0x1d, 0x1a, 0xfd, 0xf3, 0x18, 0xfc, 0xeb, 0x15, 0xf5, 0x0e, 0x0a,
0xf3, 0xf1, 0x1b, 0x05, 0x14, 0x03, 0x2d, 0x27, 0xfb, 0x18, 0x22, 0xef,
0xf6, 0x06, 0x28, 0x2b, 0xde, 0xec, 0xef, 0xe8, 0xd3, 0xfe, 0x17, 0x12,
0x01, 0x13, 0x05, 0xf7, 0x00, 0xde, 0xf3, 0xe5, 0x03, 0xfb, 0x07, 0x0b,
0xfd, 0xdc, 0xdf, 0x03, 0x0c, 0x00, 0xfa, 0x06, 0x0e, 0x02, 0x05, 0xfa,

0xfd, 0xed, 0x09, 0x0c, 0xfd, 0xfb, 0x0c, 0xf0, 0xe4, 0x04, 0xd6, 0xf3,
0x09, 0x0a, 0xf9, 0xf8, 0xe2, 0xef, 0xdf, 0xf0, 0xf8, 0x03, 0x0f, 0x20,
0xf4, 0xe3, 0xf8, 0x02, 0xe2, 0xe5, 0x25, 0x0f, 0xeb, 0xf8, 0xe9, 0xfd,
0x04, 0x0c, 0x0c, 0xfe, 0x01, 0x08, 0xfc, 0xfc, 0x1b, 0x01, 0xe5, 0x13,
0xf9, 0xe8, 0x07, 0x20, 0xfe, 0x06, 0xec, 0xfe, 0x09, 0xef, 0x14, 0x04,
0x0b, 0xf5, 0xe7, 0xff, 0x0a, 0x02, 0x09, 0xe9, 0xc4, 0x16, 0x0d, 0xe7,
0x15, 0x14, 0xf1, 0xd0, 0xec, 0xe7, 0xf0, 0xf0, 0x33, 0x05, 0xda, 0xf2,
0x0b, 0x08, 0x38, 0x01, 0x07, 0xfd, 0xd8, 0x06, 0xd9, 0xf0, 0x16, 0x1f,
0xff, 0xf7, 0xe0, 0xd8, 0xf3, 0xf7, 0x12, 0x08, 0x0e, 0x05, 0xf6, 0x03,
0xef, 0x1b, 0x12, 0xf4, 0xe8, 0x0f, 0x02, 0xfd, 0xf2, 0x16, 0x26, 0x22,
0xe0, 0x07, 0xf7, 0xe6, 0xeb, 0x16, 0x22, 0x1a, 0x0b, 0x01, 0xf5, 0xea,
0xd2, 0x22, 0x0f, 0x13, 0x15, 0x08, 0xf0, 0xfb, 0xed, 0x11, 0xf3, 0xe9,
0xff, 0xde, 0x0a, 0x18, 0x0f, 0x02, 0xfb, 0xf9, 0xfb, 0xe8, 0x12, 0x18,
0x01, 0xf4, 0xf6, 0xf8, 0xf0, 0x1f, 0x24, 0x15, 0xf5, 0x00, 0x1c, 0xf9,
0x01, 0x0a, 0x11, 0xd5, 0x01, 0x12, 0x02, 0xec, 0xfd, 0x07, 0xf2, 0xea,
0xf9, 0xff, 0xf7, 0xfb, 0x15, 0xec, 0xe5, 0x01, 0xeb, 0x05, 0xf9, 0x10,
0xfe, 0x28, 0xe5, 0x0a, 0xeb, 0x1b, 0x0e, 0xf9, 0xde, 0x02, 0x15, 0x0a,
0xff, 0xfe, 0x11, 0x24, 0x03, 0xf8, 0x00, 0x08, 0xfd, 0x0e, 0xeb, 0xf3,
0xf6, 0xf7, 0x14, 0x0e, 0xfc, 0xf5, 0xde, 0xf5, 0x9e, 0xfe, 0xff, 0xff,
0x04, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0xab, 0x01, 0x00, 0x00,
0xfa, 0xfd, 0xff, 0xff, 0xa2, 0xff, 0xff, 0xff, 0xba, 0x00, 0x00, 0x00,
0x24, 0xfc, 0xff, 0xff, 0x0f, 0x00, 0x00, 0x00, 0x54, 0x4f, 0x43, 0x4f,
0x20, 0x43, 0x6f, 0x6e, 0x76, 0x65, 0x72, 0x74, 0x65, 0x64, 0x2e, 0x00,
0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x24, 0xfb, 0xff, 0xff,
0x68, 0x01, 0x00, 0x00, 0x5c, 0x01, 0x00, 0x00, 0x50, 0x01, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0xf4, 0x00, 0x00, 0x00,

0x90, 0x00, 0x00, 0x00, 0x48, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0xce, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x09, 0x03, 0x00, 0x00, 0x00,
0x1c, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x1a, 0xff, 0xff, 0xff, 0x00, 0x00, 0x80, 0x3f, 0x01, 0x00, 0x00, 0x00,
0x09, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00,
0x00, 0x00, 0x0e, 0x00, 0x18, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x10, 0x00,
0x07, 0x00, 0x14, 0x00, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08,
0x01, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0xc4, 0xfc, 0xff, 0xff, 0x01, 0x00, 0x00, 0x00,
0x06, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00,
0x07, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0x00,
0x16, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x07, 0x00, 0x10, 0x00,
0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0x38, 0x00, 0x00, 0x00,
0x2c, 0x00, 0x00, 0x00, 0x14, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0x00,
0x14, 0x00, 0x00, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x10, 0x00, 0x07, 0x00,
0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x02, 0x00, 0x00, 0x00,
0x02, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x02, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0x00,
0x1a, 0x00, 0x08, 0x00, 0x0c, 0x00, 0x10, 0x00, 0x07, 0x00, 0x14, 0x00,
0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x11, 0x02, 0x00, 0x00, 0x00,
0x38, 0x00, 0x00, 0x00, 0x2c, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x00, 0x00,
0x00, 0x00, 0x06, 0x00, 0x08, 0x00, 0x04, 0x00, 0x06, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0xff, 0xff, 0xff, 0xff,
0x31, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00,

0x03, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x09, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x03, 0x00, 0x00, 0x00,
0x0a, 0x00, 0x00, 0x00, 0x34, 0x04, 0x00, 0x00, 0xcc, 0x03, 0x00, 0x00,
0x4c, 0x03, 0x00, 0x00, 0xdc, 0x02, 0x00, 0x00, 0x60, 0x02, 0x00, 0x00,
0x20, 0x02, 0x00, 0x00, 0xb0, 0x01, 0x00, 0x00, 0x44, 0x01, 0x00, 0x00,
0x70, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x02, 0xfc, 0xff, 0xff,
0x00, 0x00, 0x00, 0x09, 0x44, 0x00, 0x00, 0x00, 0x07, 0x00, 0x00, 0x00,
0x28, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0xf4, 0xfb, 0xff, 0xff,
0x14, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x80, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x01, 0x00, 0x00, 0x00,
0x00, 0x00, 0x80, 0x3b, 0x0e, 0x00, 0x00, 0x00, 0x6c, 0x61, 0x62, 0x65,
0x6c, 0x73, 0x5f, 0x73, 0x6f, 0x66, 0x74, 0x6d, 0x61, 0x78, 0x00, 0x00,
0x02, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x00, 0x00, 0x0e, 0x00, 0x1a, 0x00, 0x08, 0x00, 0x07, 0x00, 0x0c, 0x00,
0x10, 0x00, 0x14, 0x00, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x09,
0xb4, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x94, 0x00, 0x00, 0x00,
0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0x12, 0x00, 0x10, 0x00, 0x00, 0x00,
0x00, 0x00, 0x04, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0c, 0x00,
0x12, 0x00, 0x00, 0x00, 0x50, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00,
0x03, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x08, 0x00, 0x00, 0x00, 0x11, 0x1e, 0x23, 0x3a, 0x9e, 0xa1, 0x15, 0x39,

0x23, 0x69, 0x45, 0x3a, 0x09, 0xe4, 0xe4, 0x39, 0x65, 0xd7, 0x13, 0x3a,
0xe0, 0xb2, 0xfd, 0x39, 0x1b, 0xc1, 0x53, 0x3a, 0xc2, 0x50, 0x2d, 0x3a,
0x12, 0x00, 0x00, 0x00, 0x66, 0x69, 0x72, 0x73, 0x74, 0x5f, 0x77, 0x65,
0x69, 0x67, 0x68, 0x74, 0x73, 0x2f, 0x72, 0x65, 0x61, 0x64, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x0a, 0x00, 0x00, 0x00,
0x08, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0x3a, 0xfd, 0xff, 0xff,
0x00, 0x00, 0x00, 0x09, 0x54, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00,
0x28, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x2c, 0xfd, 0xff, 0xff,
0x14, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0xb5, 0xfa, 0xfa, 0x39, 0x1f, 0x00, 0x00, 0x00, 0x66, 0x69, 0x6e, 0x61,
0x6c, 0x5f, 0x66, 0x63, 0x5f, 0x77, 0x65, 0x69, 0x67, 0x68, 0x74, 0x73,
0x2f, 0x72, 0x65, 0x61, 0x64, 0x2f, 0x74, 0x72, 0x61, 0x6e, 0x73, 0x70,
0x6f, 0x73, 0x65, 0x00, 0x02, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0xa0, 0x0f, 0x00, 0x00, 0xa2, 0xfd, 0xff, 0xff, 0x00, 0x00, 0x00, 0x09,
0x58, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x44, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x74, 0xfe, 0xff, 0xff, 0x30, 0x00, 0x00, 0x00,
0x24, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0xf2, 0xdd, 0xbb, 0x3d,
0x01, 0x00, 0x00, 0x00, 0x32, 0xa3, 0x25, 0x41, 0x01, 0x00, 0x00, 0x00,
0xf6, 0xa0, 0x50, 0xc1, 0x05, 0x00, 0x00, 0x00, 0x61, 0x64, 0x64, 0x5f,
0x31, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x0e, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x00, 0x02,
0x2c, 0x00, 0x00, 0x00, 0x06, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00,
0x08, 0x00, 0x00, 0x00, 0x04, 0x00, 0x04, 0x00, 0x04, 0x00, 0x00, 0x00,

0x0f, 0x00, 0x00, 0x00, 0x52, 0x65, 0x73, 0x68, 0x61, 0x70, 0x65, 0x5f,
0x32, 0x2f, 0x73, 0x68, 0x61, 0x70, 0x65, 0x00, 0x01, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x4a, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x00, 0x09,
0x5c, 0x00, 0x00, 0x00, 0x05, 0x00, 0x00, 0x00, 0x44, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x1c, 0xff, 0xff, 0xff, 0x30, 0x00, 0x00, 0x00,
0x24, 0x00, 0x00, 0x00, 0x18, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x80, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0x00, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x50, 0x50, 0xd0, 0x3d,
0x01, 0x00, 0x00, 0x00, 0x00, 0x80, 0xcf, 0x41, 0x01, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x09, 0x00, 0x00, 0x00, 0x52, 0x65, 0x73, 0x68,
0x61, 0x70, 0x65, 0x5f, 0x32, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x31, 0x00, 0x00, 0x00, 0x28, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0xc2, 0xfe, 0xff, 0xff, 0x00, 0x00, 0x00, 0x09,
0x58, 0x00, 0x00, 0x00, 0x0a, 0x00, 0x00, 0x00, 0x40, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x94, 0xff, 0xff, 0xff, 0x2c, 0x00, 0x00, 0x00,
0x20, 0x00, 0x00, 0x00, 0x14, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x80, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0x01, 0x00, 0x00, 0x00, 0x50, 0x50, 0xd0, 0x3d, 0x01, 0x00, 0x00, 0x00,
0x00, 0x80, 0xcf, 0x41, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x09, 0x00, 0x00, 0x00, 0x52, 0x65, 0x73, 0x68, 0x61, 0x70, 0x65, 0x5f,
0x31, 0x00, 0x00, 0x00, 0x02, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0xa8, 0x07, 0x00, 0x00, 0x2e, 0xff, 0xff, 0xff, 0x00, 0x00, 0x00, 0x09,
0x60, 0x00, 0x00, 0x00, 0x09, 0x00, 0x00, 0x00, 0x4c, 0x00, 0x00, 0x00,
0x10, 0x00, 0x00, 0x00, 0x0c, 0x00, 0x14, 0x00, 0x04, 0x00, 0x08, 0x00,
0x0c, 0x00, 0x10, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x2c, 0x00, 0x00, 0x00,
0x20, 0x00, 0x00, 0x00, 0x14, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,

0x01, 0x00, 0x00, 0x00, 0x80, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0x01, 0x00, 0x00, 0x00, 0x3a, 0x6a, 0xac, 0x3d, 0x01, 0x00, 0x00, 0x00,
0xd0, 0xbd, 0xab, 0x41, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x52, 0x65, 0x6c, 0x75, 0x00, 0x00, 0x00, 0x00,
0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x19, 0x00, 0x00, 0x00,
0x14, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00, 0xaa, 0xff, 0xff, 0xff,
0x00, 0x00, 0x00, 0x02, 0x44, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x2c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x9c, 0xff, 0xff, 0xff,
0x18, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x01, 0x00, 0x00, 0x00, 0x96, 0x08, 0x29, 0x38, 0x0b, 0x00, 0x00, 0x00,
0x4d, 0x61, 0x74, 0x4d, 0x75, 0x6c, 0x5f, 0x62, 0x69, 0x61, 0x73, 0x00,
0x01, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0x00,
0x18, 0x00, 0x08, 0x00, 0x07, 0x00, 0x0c, 0x00, 0x10, 0x00, 0x14, 0x00,
0x0e, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x02, 0xa0, 0x00, 0x00, 0x00,
0x03, 0x00, 0x00, 0x00, 0x88, 0x00, 0x00, 0x00, 0x10, 0x00, 0x00, 0x00,
0x0c, 0x00, 0x0c, 0x00, 0x00, 0x00, 0x00, 0x00, 0x04, 0x00, 0x08, 0x00,
0x0c, 0x00, 0x00, 0x00, 0x4c, 0x00, 0x00, 0x00, 0x04, 0x00, 0x00, 0x00,
0x08, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x08, 0x00, 0x00, 0x00,
0x9a, 0xbb, 0x84, 0x38, 0x83, 0x84, 0x73, 0x37, 0x5b, 0xa3, 0xa0, 0x38,
0x16, 0x41, 0x3a, 0x38, 0xc7, 0x9a, 0x70, 0x38, 0xed, 0x70, 0x4e, 0x38,

0x54, 0x4f, 0xac, 0x38, 0xfd, /* Copyright 2023 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

=====
====*/

/* Copyright 2020 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

=====
====*/

// This is a standard TensorFlow Lite FlatBuffer model file that has been

```

// converted into a C data array, so it can be easily compiled into a
binary
// for devices that don't have a file system. It was created using the
command:
// xxd -i model.tflite > model.cc

#ifdef
TensorFlow_Lite_Micro_Examples_Micro_Speech_Micro_Features_Model_H_
#define
TensorFlow_Lite_Micro_Examples_Micro_Speech_Micro_Features_Model_H_

extern const unsigned char g_model[];
extern const int g_model_len;

#endif //
TensorFlow_Lite_Micro_Examples_Micro_Speech_Micro_Features_Model_H_

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
=====
====*/

#include "recognize_commands.h"

#include <limits>

#undef DEBUG_MICRO_SPEECH

RecognizeCommands::RecognizeCommands(int32_t average_window_duration_ms,
                                     uint8_t detection_threshold,
                                     int32_t suppression_ms,
                                     int32_t minimum_count)
    : average_window_duration_ms_(average_window_duration_ms),
      detection_threshold_(detection_threshold),

```

```

        suppression_ms_(suppression_ms),
        minimum_count_(minimum_count),
        previous_results_() {
previous_top_label_ = kCategoryLabels[0]; // silence
previous_top_label_time_ = std::numeric_limits<int32_t>::min();
}

TfLiteStatus RecognizeCommands::ProcessLatestResults(
    const TfLiteTensor* latest_results, const int32_t current_time_ms,
    const char** found_command, uint8_t* score, bool* is_new_command) {
    if ((latest_results->dims->size != 2) ||
        (latest_results->dims->data[0] != 1) ||
        (latest_results->dims->data[1] != kCategoryCount)) {
        MicroPrintf(
are "
            "The results for recognition should contain %d elements, but there
            are "
            "%d in an %d-dimensional shape",
            kCategoryCount, latest_results->dims->data[1],
            latest_results->dims->size);
        return kTfLiteError;
    }

    if (latest_results->type != kTfLiteInt8) {
        MicroPrintf(
%d",
            "The results for recognition should be int8_t elements, but are
            %d",
            latest_results->type);
        return kTfLiteError;
    }

    if ((!previous_results_.empty()) &&
        (current_time_ms < previous_results_.front().time_)) {
        MicroPrintf(
            "Results must be fed in increasing time order, but received a "
            "timestamp of %d that was earlier than the previous one of %d",
            current_time_ms, previous_results_.front().time_);
        return kTfLiteError;
    }

    // Prune any earlier results that are too old for the averaging window.
    const int64_t time_limit = current_time_ms -
average_window_duration_ms_;
    while ((!previous_results_.empty()) &&
        previous_results_.front().time_ < time_limit) {
        previous_results_.pop_front();
    }
}

```



```

}

// Add the latest results to the head of the queue.
previous_results_.push_back({current_time_ms, latest_results-
>data.int8});

// If there are too few results, assume the result will be unreliable
and
// bail.
const int64_t how_many_results = previous_results_.size();
const int64_t earliest_time = previous_results_.front().time_;
const int64_t samples_duration = current_time_ms - earliest_time;
if ((how_many_results < minimum_count_) ||
    (samples_duration < (average_window_duration_ms_ / 4))) {
    *found_command = previous_top_label_;
    *score = 0;
    *is_new_command = false;
    return kTfLiteOk;
}

// Calculate the average score across all the results in the window.
int32_t average_scores[kCategoryCount];
for (int offset = 0; offset < previous_results_.size(); ++offset) {
    PreviousResultsQueue::Result previous_result =
        previous_results_.from_front(offset);
    const int8_t* scores = previous_result.scores;
    for (int i = 0; i < kCategoryCount; ++i) {
        if (offset == 0) {
            average_scores[i] = scores[i] + 128;
        } else {
            average_scores[i] += scores[i] + 128;
        }
    }
}
for (int i = 0; i < kCategoryCount; ++i) {
    average_scores[i] /= how_many_results;
}

// Find the current highest scoring category.
int current_top_index = 0;
int32_t current_top_score = 0;
for (int i = 0; i < kCategoryCount; ++i) {
    if (average_scores[i] > current_top_score) {
        current_top_score = average_scores[i];
        current_top_index = i;
    }
}

```

```

    }
}
const char* current_top_label = kCategoryLabels[current_top_index];

// If we've recently had another label trigger, assume one that occurs
too
// soon afterwards is a bad result.
int64_t time_since_last_top;
if ((previous_top_label_ == kCategoryLabels[0]) ||
    (previous_top_label_time_ == std::numeric_limits<int32_t>::min())) {
    time_since_last_top = std::numeric_limits<int32_t>::max();
} else {
    time_since_last_top = current_time_ms - previous_top_label_time_;
}
if ((current_top_score > detection_threshold_) &&
    ((current_top_label != previous_top_label_) ||
     (time_since_last_top > suppression_ms_))) {
#ifdef DEBUG_MICRO_SPEECH
    MicroPrintf("Scores: s %d u %d y %d n %d %s -> %s",
average_scores[0],
                average_scores[1], average_scores[2], average_scores[3],
                previous_top_label_, current_top_label);
#endif // DEBUG_MICRO_SPEECH
    previous_top_label_ = current_top_label;
    previous_top_label_time_ = current_time_ms;
    *is_new_command = true;
} else {
#ifdef DEBUG_MICRO_SPEECH
    if (current_top_label != previous_top_label_) {
        MicroPrintf("#Scores: s %d u %d y %d n %d %s -> %s",
average_scores[0],
                average_scores[1], average_scores[2], average_scores[3],
                previous_top_label_, current_top_label);
        previous_top_label_ = current_top_label;
    }
#endif // DEBUG_MICRO_SPEECH
    *is_new_command = false;
}
*found_command = current_top_label;
*score = current_top_score;

return kTfLiteOk;
}

```

/* Copyright 2022 The TensorFlow Authors. All Rights Reserved.

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.

```
=====  
====*/
```

```
#ifndef TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_RECOGNIZE_COMMANDS_H_  
#define TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_RECOGNIZE_COMMANDS_H_
```

```
#include <cstdint>
```

```
#include "micro_features_micro_model_settings.h"
```

```
#include "tensorflow/lite/c/common.h"
```

```
#include "tensorflow/lite/micro/micro_log.h"
```

```
// Partial implementation of std::dequeue, just providing the  
functionality
```

```
// that's needed to keep a record of previous neural network results over  
a
```

```
// short time period, so they can be averaged together to produce a more  
// accurate overall prediction. This doesn't use any dynamic memory  
allocation
```

```
// so it's a better fit for microcontroller applications, but this does  
mean
```

```
// there are hard limits on the number of results it can store.
```

```
class PreviousResultsQueue {
```

```
public:
```

```
    PreviousResultsQueue() : front_index_(0), size_(0) {}
```

```
// Data structure that holds an inference result, and the time when it  
// was recorded.
```

```
struct Result {
```

```
    Result() : time_(0), scores() {}
```

```
    Result(int32_t time, int8_t* input_scores) : time_(time) {
```

```
        for (int i = 0; i < kCategoryCount; ++i) {
```

```
            scores[i] = input_scores[i];
```

```

    }
}
int32_t time_;
int8_t scores[kCategoryCount];
};

int size() { return size_; }
bool empty() { return size_ == 0; }
Result& front() { return results_[front_index_]; }
Result& back() {
    int back_index = front_index_ + (size_ - 1);
    if (back_index >= kMaxResults) {
        back_index -= kMaxResults;
    }
    return results_[back_index];
}

void push_back(const Result& entry) {
    if (size() >= kMaxResults) {
        MicroPrintf("Couldn't push_back latest result, too many already!");
        return;
    }
    size_ += 1;
    back() = entry;
}

Result pop_front() {
    if (size() <= 0) {
        MicroPrintf("Couldn't pop_front result, none present!");
        return Result();
    }
    Result result = front();
    front_index_ += 1;
    if (front_index_ >= kMaxResults) {
        front_index_ = 0;
    }
    size_ -= 1;
    return result;
}

// Most of the functions are duplicates of dequeue containers, but this
// is a helper that makes it easy to iterate through the contents of the
// queue.
Result& from_front(int offset) {
    if ((offset < 0) || (offset >= size_)) {

```

```

        MicroPrintf("Attempt to read beyond the end of the queue!");
        offset = size_ - 1;
    }
    int index = front_index_ + offset;
    if (index >= kMaxResults) {
        index -= kMaxResults;
    }
    return results_[index];
}

private:
    static constexpr int kMaxResults = 50;
    Result results_[kMaxResults];

    int front_index_;
    int size_;
};

// This class is designed to apply a very primitive decoding model on top
// of the
// instantaneous results from running an audio recognition model on a
// single
// window of samples. It applies smoothing over time so that noisy
// individual
// label scores are averaged, increasing the confidence that apparent
// matches
// are real.
// To use it, you should create a class object with the configuration you
// want, and then feed results from running a TensorFlow model into the
// processing method. The timestamp for each subsequent call should be
// increasing from the previous, since the class is designed to process a
// stream
// of data over time.
class RecognizeCommands {
public:
    // labels should be a list of the strings associated with each one-hot
    // score.
    // The window duration controls the smoothing. Longer durations will
    // give a
    // higher confidence that the results are correct, but may miss some
    // commands.
    // The detection threshold has a similar effect, with high values
    // increasing
    // the precision at the cost of recall. The minimum count controls how
    // many

```

```

// results need to be in the averaging window before it's seen as a
reliable
// average. This prevents erroneous results when the averaging window is
// initially being populated for example. The suppression argument
disables
// further recognitions for a set time after one has been triggered,
which can
// help reduce spurious recognitions.
explicit RecognizeCommands(int32_t average_window_duration_ms = 1000,
                           uint8_t detection_threshold = 200,
                           int32_t suppression_ms = 1500,
                           int32_t minimum_count = 3);

// Call this with the results of running a model on sample data.
TfLiteStatus ProcessLatestResults(const TfLiteTensor* latest_results,
                                  const int32_t current_time_ms,
                                  const char** found_command, uint8_t*
score,
                                  bool* is_new_command);

private:
// Configuration
int32_t average_window_duration_ms_;
uint8_t detection_threshold_;
int32_t suppression_ms_;
int32_t minimum_count_;

// Working variables
PreviousResultsQueue previous_results_;
const char* previous_top_label_;
int32_t previous_top_label_time_;
};

#endif //
TENSORFLOW_LITE_MICRO_EXAMPLES_MICRO_SPEECH_RECOGNIZE_COMMANDS_H_

{
  "data type": "audio-pcm-16khz-mono-s16",
  "delay after": 4.0,
  "test data": [
    {
      "file name": "examples/micro_speech/data/no_1000ms.wav",
      "label": "no",
      "regex": "Heard (.+) \\((\\d+\\) @\\d+ms",
      "expr": "groups[1] == label"
    }
  ]
}

```

```
    },  
    {  
      "file name": "examples/micro_speech/data/yes_1000ms.wav",  
      "label": "yes",  
      "regex": "Heard (.+) \\((\\d+\\) @\\d+ms",  
      "expr": "groups[1] == label"  
    }  
  ]  
}
```