Terminal ESP32

un appareil portable avec un écran tactile

Johan van den Brande (Belgique)

Le Terminal ESP32 d'Elecrow est un appareil portable piloté par un ESP32-3 avec un écran tactile capacitif TFT de 3,5 pouces, 480 × 320 pixels et une multitude de possibilités. Cet appareil peut être un complément intéressant à vos projets si vous avez besoin d'un écran tactile avec des possibilités d'interfaçage.

L'écran basé sur un pilote d'affichage ILI9488 a une profondeur de couleur de 16 bits, ses capacités tactiles sont gérées par un FT6236. Les deux puces sont très bien prises en charge par la communauté Arduino. Le Terminal ESP32 est logé dans une coque acrylique noire, ce qui lui donne une impression de robustesse, et certainement pas dans un boîtier en plastique rudimentaire. Un connecteur de batterie est présent, avec un circuit de charge LiPo, mais la portabilité serait meilleure si une batterie était présente dans l'appareil. Si vous voulez rendre votre projet portable, vous devrez ajouter une batterie à l'arrière. Il y a deux trous de montage M3 à l'arrière que vous pouvez utiliser pour attacher des batteries ou le fixer au mur.

Le microcontrôleur ESP32-S3

En regardant ce qui alimente l'appareil, je suis toujours étonné par l'incroyable puissance de calcul des microcontrôleurs modernes. Le terminal ESP est alimenté par un ESP32-S3 d'Espressif [2], qui contient un microcontrôleur XTensa LX7 32 bits à double cœur cadencé à 240 MHz. Comparez cela à l'Arduino UNO original avec son ATMega328 8 bits de Microchip cadencé à 16 MHz !

Le microcontrôleur dispose de 512 Ko de SRAM à bord, ainsi que de 8 Mo de PSRAM. PSRAM signifie pseudo-statique RAM, une sorte de RAM statique externe qui, dans le cas de l'ESP32-S3, est connectée au microcontrôleur via une interface SPI.
En ce qui concerne la connectivité sans fil, nous avons le Wifi 2,4 GHz (2,4 GHz 802.11 b/g/n) et le Bluetooth 5 LE. Je ne le savais pas, mais Bluetooth 5 a des capacités de longue portée, et ce module prétend les prendre en charge. Le mode longue portée étend la portée de 10 à 30 mètres à plus d'un kilomètre.

Se connecter au monde physique

Le terminal ESP32 dispose d'un connecteur de batterie et d'un chargeur LiPo embarqué. Vous pouvez utiliser le connecteur USB-C pour alimenter l'appareil, mais aussi pour charger la batterie LiPo. Il y a également un emplacement pour carte micro-SD, utile si vous souhaitez stocker quelques images ou d'autres ressources (graphiques) comme des pages web pour votre application.
Les côtés du module comportent un total de quatre ports « Crowtail » [3]. Il s'agit d'interfaces 4 fils compatibles avec Grove de Seeed Studio. Il y a un connecteur

Figure 1. Le Terminal ESP32 dans son boîtier noir affichant les données météo.

Figure 2. Les quatre ports « Crowtail » sont visibles à l'arrière.

numérique, un analogique, un série (UART) et un I²C, ce qui est suffisant pour des projets simples. Si vous avez besoin de plus, vous pouvez utiliser le port I²C comme une extension.

Programmation du terminal ESP32

Le téléchargement du micrologiciel se fait via le câble USB-C, sur une connexion série rendue possible par un convertisseur USB-série CH340 [4]. Je n'ai pas eu de chance en essayant de faire fonctionner cela avec mon MacBook, pour lequel j'ai essayé d'installer le pilote officiel. Heureusement, j'ai toujours mon vieil ordinateur portable Linux, qui tourne sous Ubuntu et qui fonctionne parfaitement !

La première chose que je voulais essayer, c'était d'exécuter une version de Python sur l'appareil. Cela s'est avéré difficile. Bien que la documentation indique qu'il peut être programmé avec Python et MicroPython, je n'ai pas trouvé de micrologiciel téléchargeable prêt à l'emploi.

En lisant leur site web, on y trouve de nombreux exemples et un tutoriel pour Arduino [5], et c'est donc ce que j'ai commencé à utiliser pour mes applications de démonstration.

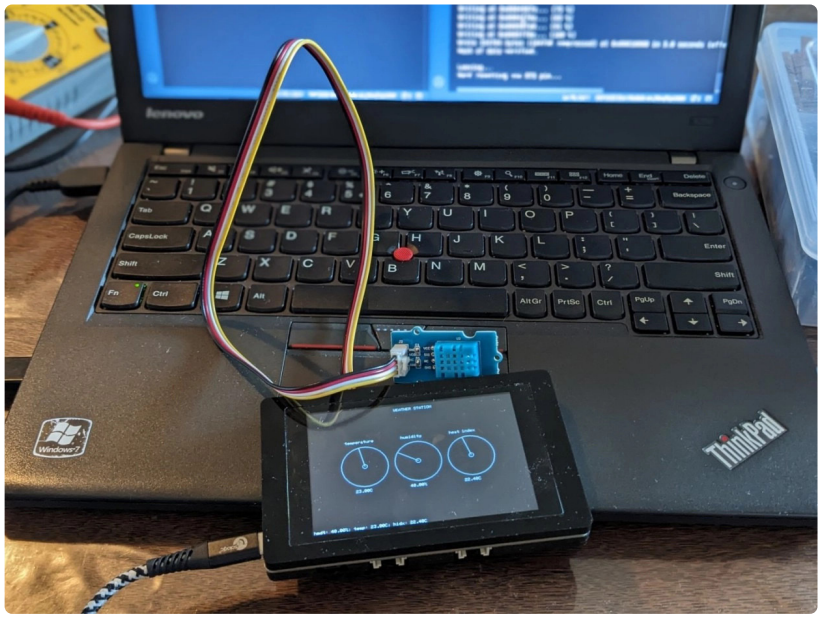

Figure 3. Ajoutez un capteur DHT-11 et un peu de code et vous obtiendrez une station météorologique simple.

Conception de l'interface utilisateur

Selon la page web du Terminal ESP32, l'appareil est certifié LVGL [6][7]. LVGL signifie *Light and Versatile Graphics Library*, qui est une bibliothèque graphique embarquée open source permettant de créer des interfaces utilisateur pour des microcontrôleurs et des affichages divers. Bien que la bibliothèque soit open source en elle-même, un éditeur de mise en page (par glisser-déposer) à logiciel propriétaire, Squareline Studio [8], est également disponible.

J'ai installé la version d'essai et bien qu'elle soit un peu déroutante à première vue, j'ai rapidement réussi à dessiner une interface utilisateur simple pour le projet de servomoteur ci-dessous. Si votre objectif est de créer des interfaces utilisateur pour des systèmes embarqués, cela vaut la peine d'investir un peu de temps dans l'apprentissage de ce produit.

Deux petits projets

J'ai décidé de créer deux projets d'exemple. Le premier projet est une petite station météorologique, où nous dessinons notre propre interface utilisateur en utilisant les primitives ligne et cercle. Le second projet est un exemple où nous contrôlons un servo à partir d'une interface utilisateur conçue avec Squareline Studio.

Station météo

Pour la station météorologique, nous utilisons un capteur DHT-11, qui peut détecter la température et l'humidité. Il possède une interface numérique à un fil et est bien supporté par Arduino. Pour ce projet, j'ai choisi la bibliothèque *Adafruit DHT [9]*. Elle fait partie d'un ensemble de bibliothèques de capteurs partageant un code commun, et par conséquent nous devons également installer la bibliothèque *Adafruit Unified Sensor Driver [10]*.

Dans ce projet, j'ai décidé de créer ma propre interface utilisateur à partir de rien, en utilisant des primitives pour imprimer du texte et créer quelques lignes graphiques. Pour cela, j'ai utilisé la bibliothèque graphique *LCD* et *e-Ink* de LovyanGFX [11][12], une autre bibliothèque graphique open-source.

Après avoir initialisé l'écran, pour lequel j'ai simplement copié un des exemples du site de Terminal ESP32, nous lisons la mesure du capteur DHT-11. Celui-ci nous donne trois valeurs : la température, l'humidité et l'indice de chaleur. L'indice de chaleur est une température ajustée par l'humidité réelle. Ces trois valeurs sont affichées dans un cadran simple, dessiné à l'aide de deux cercles et d'une ligne comme pointeur. L'écran est mis à jour toutes les deux secondes avec une nouvelle valeur.

Contrôleur de servo

Pour le contrôleur de servo, j'ai voulu essayer l'outil Squareline Studio pour créer l'interface utilisateur. Un curseur en forme d'arc est le seul widget utilisé. La plage du curseur est réglée de 0 à 180, ce qui correspond à l'angle du servo.

Lorsque vous concevez une interface utilisateur pour l'ESP32 à l'aide de cet outil, vous devez commencer par le modèle *Arduino with TFT_eSPI* et définir la profondeur de couleur à 16 bits et la résolution à 480 × 320 pixels. Après avoir exporté le code généré, vous devrez le décompresser dans un répertoire nommé *ui* dans votre dossier *libraries* d'Arduino. Normalement, vous trouverez ce répertoire *libraries* dans le même répertoire que celui où se trouvent vos croquis Arduino… Il m'a fallu un certain temps pour m'en rendre compte.

Le servo est piloté par la bibliothèque *ESP32Servo*. La bibliothèque standard *Arduino servo* ne fonctionne pas avec l'ESP32.

Le code est assez basique, dans la fonction `loop`, nous lisons l'angle du curseur d'arc, qui renvoie un nombre entre 0 et 180, puis nous envoyons cette valeur à la fonction `servo.write`.

Vous pouvez trouver le code source des deux projets dans la section de téléchargement de cet article. Je n'ai pas inclus le dossier *libraries* dans le téléchargement, car il est (beaucoup) trop volumineux. Installer les dépendances soi-même n'est pas compliqué.

Conclusion

Le Terminal ESP32 d'Elecrow n'est pas aussi facile à apprivoiser que je l'espérais, mais il a beaucoup de potentiel. Si vous voulez investir du temps pour apprendre la bibliothèque LVGL ou une autre bibliothèque d'interface utilisateur, et que vous êtes à l'aise avec Arduino, alors avec le Terminal ESP32 vous pouvez ajouter une interface utilisateur avec un contrôle tactile à votre prochain projet.

Cela aurait été plus facile si le micrologiciel MicroPython [15] avait été disponible en téléchargement sur le site du produit. J'ai cherché un peu sur internet, mais je n'ai rien trouvé. Il est certainement possible de compiler votre propre micrologiciel, et c'est peut-être une bonne idée pour votre prochain projet ?

VF : Laurent Rauber — 230755-04

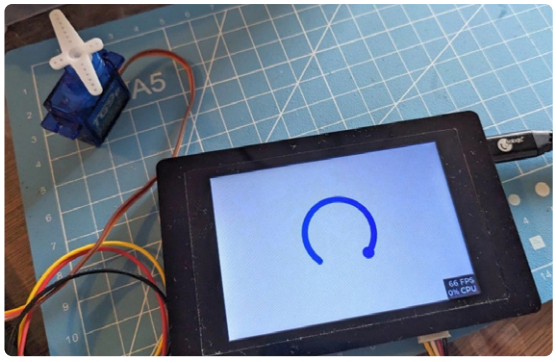

Figure 4. Le curseur circulaire contrôle la position du servo.

Questions ou commentaires ?

Contactez Elektor (redaction@elektor.fr).

Produits

- **ESP Terminal (carte de développement ESP32-S3 avec écran tactile TFT capacitif de 3,5 pouces)** www.elektor.fr/20526

LIENS

[1] Terminal ESP32 d'Elecrow : https://www.elektor.fr/esp-terminal-esp32-s3-based-development-board-with-3-5-capacitive-tft-touch-display
[2] Fiche technique de l'ESP32-S3 : https://www.espressif.com/sites/default/files/documentation/esp32-s3_datasheet_en.pdf
[3] Ports « Crowtail » : https://www.elecrow.com/wiki/index.php?title=Main_Page#Crowtail
[4] CH340 convertisseur USB-série : https://cdn.sparkfun.com/datasheets/Dev/Arduino/Other/CH340DS1.PDF
[5] Tutoriel pour Arduino : https://www.elecrow.com/wiki/index.php?title=Lesson01_Introducing_the_ESP32_Display_series_and_environment_configuration
[6] LVGL : https://lvgl.io/
[7] Documentation LVGL : https://docs.lvgl.io/latest/en/html/index.html
[8] Squareline Studio : https://squareline.io/
[9] Bibliothèque DHT-11 : https://github.com/adafruit/DHT-sensor-library
[10] Unified Sensor Library : https://github.com/adafruit/Adafruit_Sensor
[11] LovyanGFX LCD and e-Ink graphics library: https://github.com/ropg/LovyanGFX
[12] Documentation LovyanGFX : https://lovyangfx.readthedocs.io/en/latest/index.html
[13] Bibliothèque ESP32Servo : https://www.arduino.cc/reference/en/libraries/esp32servo/
[14] Téléchargement du micrologiciel : https://www.elektormagazine.fr/review/terminal-esp32-test
[15] « MicroPython pour l'ESP32 et ses copains - Partie 1 : installation et premiers programmes », Günter Spanner, 2021: https://www.elektormagazine.fr/articles/micropython-pour-l-esp32-et-ses-copains-partie-1