Come utilizzare TrustZone per proteggere i dispositivi IoT con minimi livelli di complessità hardware e di costi

Di Jacob Beningo
Contributo di Editori nordamericani di DigiKey
2020-07-23

I dispositivi IoT alla periferia della rete richiedono l'adozione di misure di sicurezza aggiuntive rispetto a quelle tradizionalmente necessarie per i prodotti embedded. Conoscere e lavorare sugli aspetti di sicurezza è spesso difficile e gli sviluppatori preparati nel campo dell'ingegneria elettrica non hanno esperienza nella crittografia o nella sicurezza dei sistemi.

Oggi, in un ciclo di sviluppo il tempo è poco e i budget sono scarni e gli sviluppatori non possono permettersi di cominciare da zero per diventare in fretta esperti di sicurezza. Invece, possono sfruttare le soluzioni di sicurezza esistenti e adattarle alle loro specifiche esigenze.

Questo articolo presenterà TrustZone di Arm®, un'interessante soluzione che sta prendendo piede tra i progettisti di sistemi basati su microcontroller. A titolo di esempio, l'articolo esaminerà la serie STM32L5 di microcontroller di STMicroelectronics che supporta TrustZone e mostrerà come iniziare a utilizzare TrustZone utilizzando i kit di sviluppo associati.

Sicurezza attraverso l'isolamento

L'elemento centrale fondamentale per un sistema embedded sicuro è la sicurezza attraverso l'isolamento. L'idea è che importanti risorse di dati come chiavi private, dati degli utenti, funzioni sicure e così via devono essere isolate dai dati e dalle funzioni generiche, come gli elementi dell'interfaccia utente grafica o il sistema operativo in tempo reale (RTOS). Sebbene esistano metodi per isolare il software, gli esperti di sicurezza concordano sul fatto che un sistema embedded deve utilizzare la sicurezza attraverso l'isolamento basato sull'hardware.

Vi sono diversi modi di utilizzare l'hardware per creare questo isolamento, ad esempio attraverso un microcontroller e un processore di sicurezza o mediante un processore multicore dove un core è dedicato all'elaborazione sicura. I più recenti processori Arm Cortex®-M23, Cortex-M33 e Cortex-M55 supportano una funzione opzionale di isolamento basata sull'hardware, nota come TrustZone.

Che cos'è Arm TrustZone?

TrustZone è un meccanismo hardware implementato nei microcontroller single core che divide l'ambiente di esecuzione in memoria, periferiche e funzioni sicure e non sicure. Ogni ambiente di esecuzione contiene anche un'unità di protezione di memoria (MPU) che può essere impiegata per isolare ulteriormente le regioni di memoria e fornire "più strati a cipolla" quali deterrenti ai potenziali aggressori che tentano di accedere alle risorse di dati.

In generale, uno sviluppatore embedded partiziona il proprio sistema in almeno due progetti: il progetto di esecuzione non sicura, spesso indicato come progetto utente, e il progetto di esecuzione sicura, spesso indicato come progetto firmware. Un microcontroller che abilita TrustZone si avvia nello stato sicuro e avvia il sistema prima di passare allo stato non sicuro per eseguire l'applicazione utente (Figura 1).

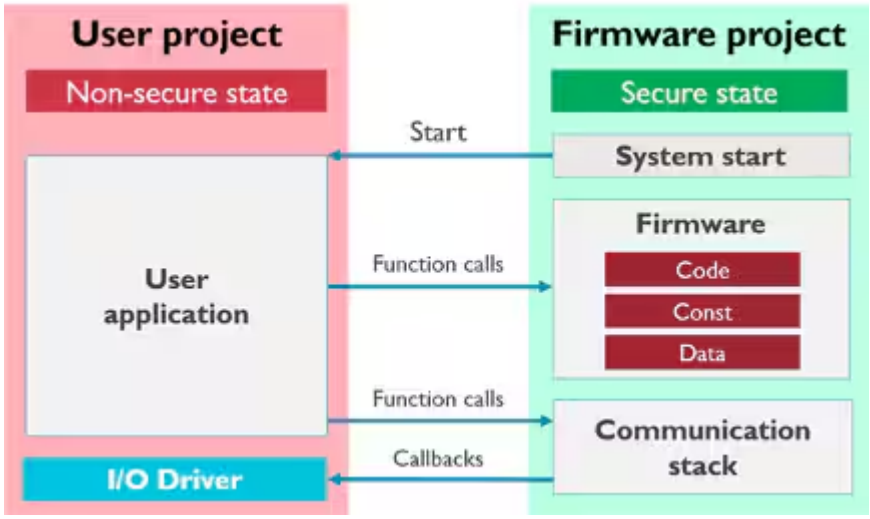

Figura 1: I progetti TrustZone attuano l'isolamento attraverso un meccanismo hardware che divide il software embedded in un progetto utente (non sicuro) e un progetto firmware (sicuro). (Immagine per gentile concessione di Arm)

Il progetto utente può accedere alle funzioni sicure solo attraverso un gateway sicuro, creato tra il progetto firmware e il progetto utente, e non può accedere alle posizioni di memoria sicure senza attivare un'eccezione.

Selezione di una scheda di sviluppo abilitata per TrustZone

Il modo più semplice per iniziare a capire TrustZone è semplicemente di iniziare a lavorarci. A tale scopo, uno sviluppatore deve prima selezionare una scheda di sviluppo. Vi sono varie scelte tra le schede di sviluppo, di vari fornitori di microcontroller, ma attenzione: non tutte implementano TrustZone allo stesso modo, e questo può rendere il tutto un po' più complicato.

Un buon esempio di scheda di sviluppo per iniziare a lavorare con TrustZone è il kit Discovery STM32L562E di STMicroelectronics (Figura 2).

Figura 2: Il kit Discovery STM32L562E include numerosi sensori, Bluetooth e una scheda di espansione I/O che facilita lo sviluppo di applicazioni TrustZone. (Immagine per gentile concessione di STMicroelectronics)

Il kit viene fornito con molte funzioni di supporto utili quando si lavora con TrustZone per la prima volta. Ad esempio, il kit di sviluppo include un modulo LCD TFT da 1,54 pollici, 240 x 240 pixel comprendente un pannello di controllo touchscreen, un modulo Bluetooth Low Energy v4.1, un accelerometro e giroscopio 3D iNEMO e un STLINK-V3E, oltre a molte altre funzioni per l'espansione di I/O e periferiche.

Una seconda scheda di sviluppo utile per muovere i primi passi con TrustZone è la scheda NUCLEO-L552ZE-Q di STMicroelectronics (Figura 3).

Figura 3: La scheda di sviluppo STM32L552ZE-Q NUCLEO fornisce un processore abilitato per TrustZone, ST-LinkV3 e basette di espansione per attività di sviluppo personalizzate. (Immagine per gentile concessione di STMicroelectronics)

A differenza del kit Discovery STM32L562E, NUCLEO-L552ZE-Q è una scheda di sviluppo di base che include ST-LinkV3, il microcontroller STM32L552VET6, porte di espansione e un LED. Questa scheda è ottima per gli sviluppatori interessati ad armeggiare con TrustZone per iniziare a integrare i propri componenti hardware il più rapidamente possibile.

Anche se NUCLEO-L552ZE-Q non è a piena funzionalità, STM32L552VET6 è piuttosto impressionante. Si tratta di un processore Arm Cortex-M33 con un'unità a virgola mobile (FPU), fino a 512 kB di memoria flash e 256 kB di SRAM. Include diverse funzioni di sicurezza aggiuntive oltre a TrustZone, come una radice di attendibilità con un'unica voce di avvio, installazione sicura del firmware e supporto per l'aggiornamento sicuro del firmware con firmware attendibile per Cortex-M (TF-M).

I processori di entrambe le schede di sviluppo includono un'unità TrustZone SAU (Security Arbitration Unit) utilizzata per impostare le memorie e periferiche che saranno protette da TrustZone. La SAU manca nelle implementazioni TrustZone di alcuni fornitori di microcontroller. Questo non è necessariamente un problema, ma richiede una procedura diversa per la sua impostazione.

Iniziare la prima applicazione basata su TrustZone

La messa in funzione di una delle schede di sviluppo di STMicroelectronics richiede alcuni passaggi e pacchetti software. Per prima cosa, uno sviluppatore vorrà scaricare STM32CubeIDE, che fornisce il compilatore, i pacchetti di microcontroller e l'IDE per lo sviluppo di applicazioni e tutte le note applicative associate, ivi compresa STM AN5394.

L'utilizzo di un progetto di esempio TrustZone esistente è il modo più rapido per creare un'applicazione funzionante. Sono inclusi diversi progetti nel pacchetto software STM32Cube_FW_L5. Questo software viene scaricato come parte del software STM32CubeL5. Una volta scaricato, gli sviluppatori possono importare il progetto TrustZoneEnabled che si trova in un percorso di directory simile a questo:

STM32Cube_FW_L5_V1.2.0\STM32Cube_FW_L5_V1.2.0\Projects\STM32L552E-EV\Templates\TrustZoneEnabled

Una volta importato che il progetto, lo sviluppatore può vederne la struttura gerarchica che divide l'applicazione in due parti, una sicura e una non sicura (Figura 4).

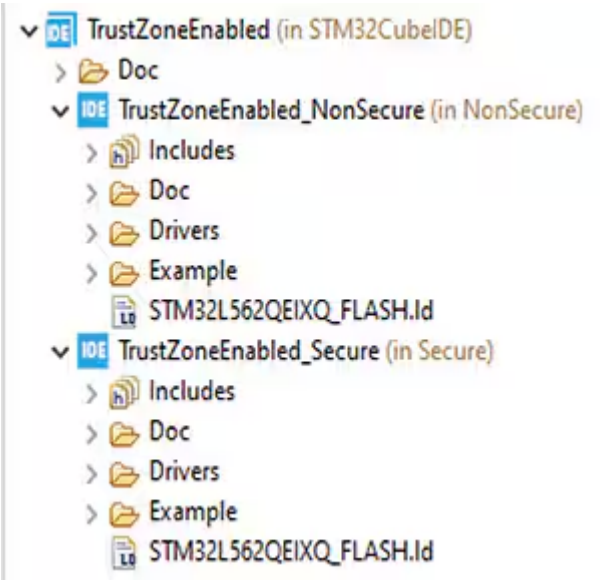

Figura 4: Il progetto TrustZone è implementato in una struttura gerarchica con due progetti, uno sicuro e uno non sicuro. (Immagine per gentile concessione di Beningo Embedded Group)

Molti dettagli possono essere esaminati in questi progetti. AN5394 è in grado di compilare parecchi dettagli, mentre il file readme.txt nella cartella Doc di ogni progetto può spiegare i dettagli relativi ai progetti sicuri e non sicuri. Ai fini di questo articolo, esamineremo i concetti più importanti relativi a TrustZone. In particolare, come si configura TrustZone. La configurazione è reperibile nel file partition_stm32L562xx.h qua:

C:\STM32Cube_FW_L5_V1.2.0\Projects\STM32L562E-DK\Templates\TrustZoneEnabled\Secure\Inc

Questo file contiene le impostazioni per la SAU. Ad esempio, la Figura 5 mostra le impostazioni per la regione SAU 0, attualmente configurata per l'esecuzione sicura. La Figura 6 mostra invece la regione SAU 1 configurata come non sicura.

```
68 /*
69 //   <e>Initialize SAU Region 0
70 //   <i> Setup SAU Region 0 memory attributes
71 */
72 #define SAU_INIT_REGION0    1
73
74 /*
75 //     <o>Start Address <0-0xFFFFFFE0>
76 */
77 #define SAU_INIT_START0     0x0C03E000      /* start address of SAU region 0 */
78
79 /*
80 //     <o>End Address <0x1F-0xFFFFFFFF>
81 */
82 #define SAU_INIT_END0       0x0C03FFFF      /* end address of SAU region 0 */
83
84 /*
85 //     <o>Region is
86 //         <0=>Non-Secure
87 //         <1=>Secure, Non-Secure Callable
88 */
89 #define SAU_INIT_NSC0       1
```

Figura 5: La SAU è utilizzata per configurare le regioni di memoria sicure e non sicure. Il codice sopra riportato mostra come la regione SAU 0 sia configurata per l'esecuzione sicura. (Immagine per gentile concessione di Beningo Embedded Group)

```
95 //   <e>Initialize SAU Region 1
96 //   <i> Setup SAU Region 1 memory attributes
97 */
98 #define SAU_INIT_REGION1    1
99
100 /*
101 //     <o>Start Address <0-0xFFFFFFE0>
102 */
103 #define SAU_INIT_START1     0x08040000      /* start address of SAU region 1 */
104
105 /*
106 //     <o>End Address <0x1F-0xFFFFFFFF>
107 */
108 #define SAU_INIT_END1       0x0807FFFF      /* end address of SAU region 1 */
109
110 /*
111 //     <o>Region is
112 //         <0=>Non-Secure
113 //         <1=>Secure, Non-Secure Callable
114 */
115 #define SAU_INIT_NSC1       0
```

Figura 6: La SAU è utilizzata per configurare le regioni di memoria sicure e non sicure. Il codice sopra riportato mostra come la regione SAU 1 sia configurata per l'esecuzione non sicura. (Immagine per gentile concessione di Beningo Embedded Group)

Lo sviluppatore deciderà quali regioni dovranno essere sicure e non sicure e utilizzerà il file della partizione per configurare la SAU. Queste impostazioni non garantiscono però l'abilitazione di TrustZone! Quando programma un'applicazione basata su TrustZone sul target, per abilitare TrustZone lo sviluppatore deve impostare il byte dell'opzione TZ su 1. Questo per abilitare TrustZone durante l'avvio in modo che la configurazione SAU possa essere letta e utilizzata.

Suggerimenti e consigli per lavorare con TrustZone

Implementare TrustZone non è difficile ma gli sviluppatori devono pensare al progetto della loro applicazione in modo un po' diverso. Ecco alcuni consigli per iniziare:

- Non sarà necessario proteggere tutti i dati. Identificare le risorse di dati critici da proteggere.
- Sfruttare i framework di sicurezza esistenti come CMSIS-Zone e Trusted Firmware per Cortex-M (TF-M) per accelerare lo sviluppo.
- Esaminare attentamente le potenziali minacce al dispositivo e selezionare un microcontroller che supporti soluzioni hardware e software per la protezione da tali minacce.
- TrustZone fornisce un unico strato di isolamento. Sfruttare le MPU e altri meccanismi hardware per creare più livelli di isolamento basati sull'hardware.
- Identificare gli elementi di codice sicuri e non sicuri durante la fase di definizione dell'architettura, non durante l'implementazione.

Seguendo questi consigli, gli sviluppatori risparmieranno parecchio tempo ed eviteranno grattacapi nell'implementazione delle funzioni di sicurezza nei loro dispositivi IoT.

Conclusione

TrustZone è uno strumento importante a disposizione degli sviluppatori di IoT desiderosi di mettere in sicurezza i loro dispositivi e proteggere le risorse di dati. Le soluzioni sicure possono essere implementate in diversi modi, ma come abbiamo visto, TrustZone si propone come una soluzione single core che fornisce un modello di sviluppo software tradizionale. L'unica differenza è che gli sviluppatori devono iniziare a pensare in termini di componenti, dati e thread sicuri e non sicuri.