

Emulation STM32L Discovery avec Qemu

Clavelin Aurélien, Eid Timothée, Mercier Michaël

Plan

① STM32L Discovery

② Qemu

③ Travail réalisé

④ Démonstration

⑤ Conclusion

STM32L Discovery

Présentation
Caractéristiques
Utilisation

Qemu

Travail réalisé

Démonstration

Conclusion

Plan

① STM32L Discovery

② Qemu

③ Travail réalisé

④ Démonstration

⑤ Conclusion

Présentation

- Découverte du STM32L
- Très basse consommation
- Carte embarquée
- Beaucoup de périphériques
- Interface de debugage



Caractéristiques

Microcontrôleur

- Adresse 32 bits
- Mémoire
 - Flash : 128 Ko
 - RAM : 16 Ko
- Others
 - RTC (Real Time Clock)
 - USART (Universal Asynchrone/Synchrone Receiver/Transmitter)
 - I2C (Inter Integrated Circuit)
 - SPI (Serial peripheral interface)
 - ADC (Analog-Digital Convertor)
 - DAC (Digital-Analog Convertor)
 - Comparateurs
 - ...

STM32L Discovery

Présentation

Caractéristiques

Utilisation

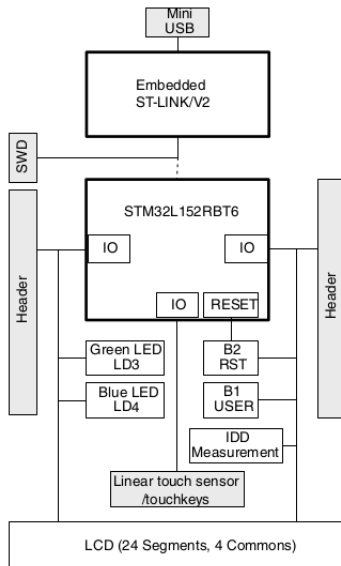
Qemu

Travail réalisé

Démonstration

Conclusion

Caractéristiques



STM32L Discovery

Présentation
Caractéristiques
Utilisation

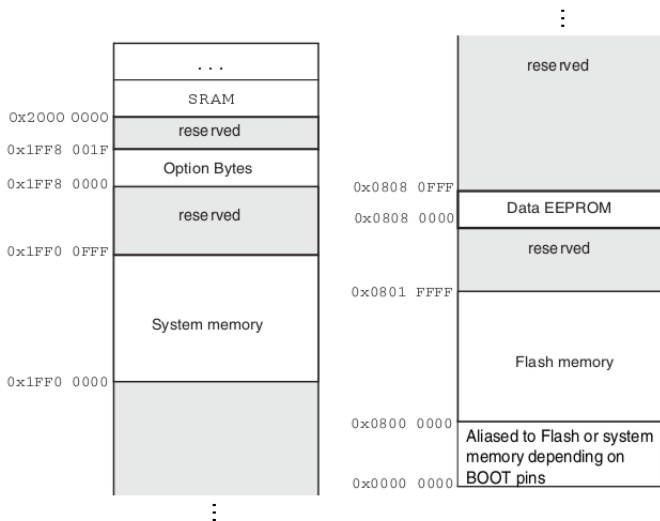
Qemu

Travail réalisé

Démonstration

Conclusion

Structure de la mémoire

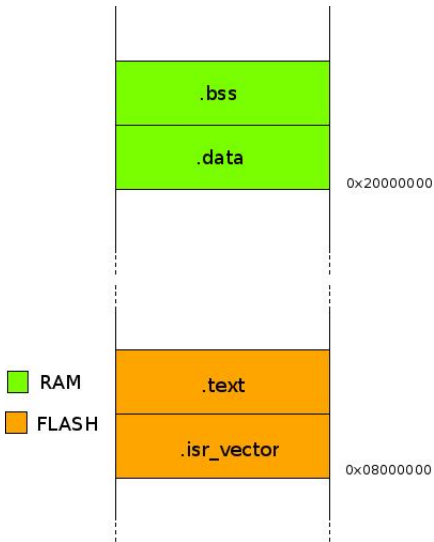


Structure d'un programme

Pour créer un executable ELF

- Code source
- Startup program
- Linker

Structure d'un programme



Chargement sur la carte

Prérequis

- Suite de cross compilation (arm-none-eabi)
- Utilitaire texane-stlink

Procédure de chargement

- Lancement de l'utilitaire texane-stlink
 - # ./st-util -p 4242
- Chargement du programme avec GDB
 - # arm-none-eabi-gdb
 - (gdb) target extended-remote :4242
 - (gdb) load ./programme.elf

Plan

① STM32L Discovery

② Qemu

③ Travail réalisé

④ Démonstration

⑤ Conclusion

STM32L Discovery

Qemu

Présentation

Utilisation

Structure

Travail réalisé

Démonstration

Conclusion



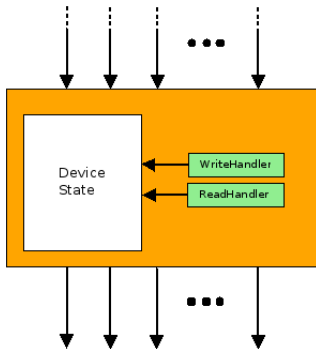
- Emulation complète
 - Intel
 - ARM
 - PowerPC
 - Sparc
 - ...
- VMWare, Virtualbox, ...

Emulation d'un programme

Procédure de chargement

- Compilation du programme utilisateur
- Lancement de Qemu avec le programme émulé

Composant



- Etat du composant (Device State)
- Interfaces d'entrées (fils)
- Interfaces de sorties (fils)
- Gestion d'une zone mémoire déléguée
- Enregistré dans Qemu : Accessible pour les machines

STM32L Discovery

Qemu

Présentation

Utilisation

Structure

Travail réalisé

Démonstration

Conclusion

- Qemu met à disposition des machines
 - [Syborg](#) Plateforme virtuelle Symbian
 - [n800](#) Nokia N800
 - [lm3s6965evb](#) Stellaris LM3S6965EVB
 - [stm32l152rbt6](#) STM32L Discovery (Travail réalisé)
- Assemblage de composants
- Communication avec des CharDev

STM32L Discovery

Qemu

Présentation

Utilisation

Structure

Travail réalisé

Démonstration

Conclusion

- Interface de communication
 - Socket
 - Clavier
 - Port série
 - ...
- Bidirectionnelle
- Fonctionnement
 - Initialisation au lancement de Qemu
 - Connexion à l'initialisation de la machine
 - Fonction d'envoi
 - Handler de réception

Plan

① STM32L Discovery

② Qemu

③ Travail réalisé

④ Démonstration

⑤ Conclusion

GPIO - Description

- General Purpose Input/Output
- Entrée/Sortie pour usage général
- Paramétrable grâce à des registres de configuration
- Modes de fonctionnement
 - Implémentés
 - Output (Push/Pull)
 - Input (Push/Pull)
 - Non implémentés
 - Output (Open-drain)
 - Input (Open-drain)
 - Alternate function

GPIO - Implémentation

- Registres représentés par une structure

```
4
5 typedef struct {
6     SysBusDevice busdev;
7
8     /* Registres GPIO (Reference Manual p119) */
9     uint32_t mode; /* Mode */
10    uint16_t otype; /* Output type */
```

- R/W en mémoire grâce aux fonctions handlers

```
74 static uint32_t stm32_gpio_read(void *opaque, target_phys_addr_t offset) {
75     stm32_gpio_state *s = (stm32_gpio_state *) opaque;
76
77     switch (offset) {
78         case 0x00: /* Mode */
79             return s->mode;
80         case 0x04: /* oType */
81             return s->otype;
82         case 0x08: /* oSpeed */
```

STM32L Discovery

Qemu

Travail réalisé

GPIO

LED

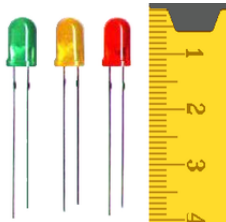
Bouton

Machine

Différences avec la carte
physique

Démonstration

Conclusion



- Connexion à une sortie GPIO
- Envoi de l'état au CharDev quand la sortie GPIO change d'état

Bouton

STM32L Discovery

Qemu

Travail réalisé

GPIO

LED

Bouton

Machine

Différences avec la carte
physique

Démonstration

Conclusion



- Connexion à une entrée GPIO
- Envoi d'un signal au GPIO quand il reçoit un signal du chardev

Clavelin Aurélien, Eid
Timothée, Mercier
Michaël

Machine

STM32L Discovery

Qemu

Travail réalisé

GPIO

LED

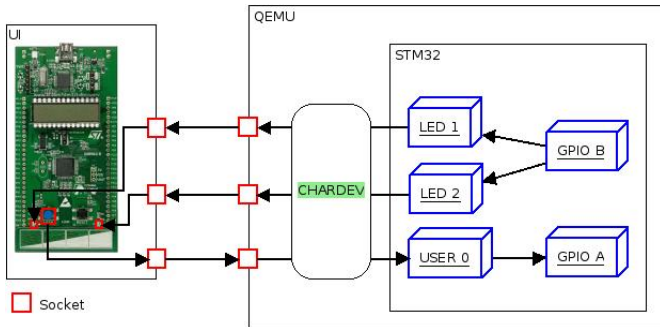
Bouton

Machine

Différences avec la carte
physique

Démonstration

Conclusion



RCC

Realtime Clock Control

Procédure d'écriture sur une pin GPIO

- Parametrage via les registres
- Ecriture dans le registre "Output data"
- La valeur sera recopiée sur la pin au prochain tick de la RCC

Problème

Le module RCC n'a pas été implémenté dans Qemu

Solution adoptée

Notre implémentation des GPIO recopie la valeur sur la pin sans attendre le tick de la RCC

Zones d'adresse

Mapping d'adresses

- Zone de boot pouvant être remappée en fonction des jumpers sur
 - SRAM : Pour les tests
 - Flash : Pour écrire un programme persistant

Problème

Ce remappage n'existe pas sur Qemu

Solution adoptée

Création de 2 version de programme

- Qemu : le programme est "nativement" logé aux adresses de boot (0x0000 0000)
- Hardware : le programme est logé en flash (0x0800 0000) puis mappé matériellement sur les adresses de boot (0x0000 0000)

Plan

1 STM32L Discovery

2 Qemu

3 Travail réalisé

4 Démonstration

5 Conclusion

GPIO Output

Déroulement

- Initialisation des GPIO
 - Le registre Mode
 - Activation de la RCC pour le GPIO_B (LED)
- While(1)
 - Switch_LED_ON()
 - Ecrire dans le registre de sortie
 - Delay()
 - Switch_LED_OFF()
 - Ecrire dans le registre de sortie
 - Delay()

GPIO Input + Attente active

Limite

Nous n'avons pas implémenté les interruptions processeur pour pouvoir utiliser des handlers d'interruption. Ce programme réalise donc une attente active

Déroulement

- Initialisation
 - GPIO_B en sortie, GPIO_A (Bouton) en entrée
 - Activation de la RCC pour les GPIO
- While(1)
 - Si l'état a changé
 - Switch_LED()

Plan

① STM32L Discovery

② Qemu

③ Travail réalisé

④ Démonstration

⑤ Conclusion

Prochaines évolutions

- Gestion des interruptions materielles
- Implémentation des protocoles (Alternate function)
- Implémentation des composants (Ecran, ADC, ...)
- Implémentation du composant RCC et des Timers

Rétrospective

Points négatifs



Points positifs



Questions

STM32L Discovery

Qemu

Travail réalisé

Démonstration

Conclusion

Prochaines évolutions

Rétrospective

Questions

