102685 Arquitectura de Computadors i Perifèrics
PRÀCTICA Nº 4 CURS 2020/2021

ENTRADA/SORTIDA UTILITZANT LES ESTRUCTURES DE DADES QUE OFEREIX L'ENTORN CODEWARIOR

Objectius de la pràctica

En aquesta sessió de pràctiques tractarem els següents temes:
1) Com accedir als registres de E/S de la placa, utilitzant les estructures de dades que ofereix l'entorn Codewarior.
2) Configurar els dispositius de E/S integrats en la placa, per realitzar un conjunt d'operacions de E/S utilitzant la tècnica d'espera programada (pooling).

Estructures de dades que ofereix Codewarior per operar amb l'E/S

La placa **FRDM-KL25Z** ofereix un conjunt de dispositius d'E/S, alguns dels quals són:

- Un control lliscant capacitiu tàctil (touch slider).
- Un led amb tres colors (RGB).
- Un acceleròmetre.
- Una interfície USB.

Per configurar i operar amb aquests dispositius, el sistema ofereix un conjunt de registres d'E/S (que formen l'espai d'E/S). Per accedir a aquests registres, el processador Cortex M0 segueix el sistema de memòria mapejada (memory mapped), amb el qual no es diferencia la manera de adreçar la memòria principal i la de l'espai d'E/S. Això vol dir que cadascun dels registres d'E/S té una adreça única, que permet el seu accés com si es tractés d'una posició de la memòria principal. Aquest fet implica que en el processador Cortex M0, no existeixen instruccions assemblador per accedir a l'espai d'E/S, com les IN i OUT d'Intel, sinó que s'utilitzen les d'accés a memòria com LDR i STR.
Normalment, els sistemes ofereixen un conjunt predefinit d'adreces per l'Espai d'E/S i unes altres per la memòria principal. En el cas de la placa **FRDM-KL25Z,** les adreces destinades a l'espai d'E/S comencen a l'adreça **0x4003B000.**
Com es va comentar en la pràctica anterior, els entorns reals que es basen, entre d'altres, en la placa **FRDM-KL25Z**, utilitzen els llenguatges d'alt nivell (com C) per programar els codis fonts dels seus executables. Per facilitar la programació de codis font que treballen amb l'espai E/S, l'entorn de programació **CodeWarrior** (igual que d'altres) ofereix un fitxer de capçalera de C per la placa **FRDM-KL25Z.** Aquest fitxer s'anomena **MKL25Z4.h** (situat en el directori *Project_Headers* del projecte) i en ell es defineixen unes estructures de dades i macros que permeten operar amb els registres d'E/S.
El fitxer **MKL25Z4.h,** descriu els registres d'E/S agrupats pel dispositiu a qui pertanyen i els dispositius estan ordenats alfabèticament pel seu nom: Primer es defineixen el registres de l'ADC, després els del BP i així fins a arribar al darrer dispositiu, que es el USB.
Les dues estructures de dades més importants que defineixen els registres d'E/S de cada dispositiu són:

a) typedef struct DISPOSITIU_MemMap {
/*definició dels registres del dispositiu*/
}volatile *DISPOSITIU_MemMapPtr;

b) #define DISPOSITIU_BASE_PTR ((DISPOSITIU_MemMapPtr) direcció origen)

A on ***DISPOSITIU*** identifica cadascun dels diferents dispositius de la placa **FRDM-KL25Z** (USB, ADC, PORTB, etc) i ***direcció origen*** defineix l'adreça a on comencen els registres d'E/S del dispositiu en qüestió. Per exemple, pel dispositiu del touch (TSI), que utilitzem en aquesta sessió de pràctiques, la definició d'aquestes dues estructures es:

```
/** TSI - Peripheral register structure */
typedef struct TSI_MemMap {
  uint32_t GENCS;          /**< TSI General Control and Status Register, offset: 0x0 */
  uint32_t DATA;           /**< TSI DATA Register, offset: 0x4 */
  uint32_t TSHD;           /**< TSI Threshold Register, offset: 0x8 */
} volatile *TSI_MemMapPtr;

#define TSI0_BASE_PTR       ((TSI_MemMapPtr) 0x40045000u)
```

TSI0_BASE_PTR defineix un apuntador a l'estructura del tipus **TSI_MemMap,** sent l'adreça a on apunta la **0x40045000,** que es precisament l'adreça predefinida pel sistema per al primer registre d'E/S (el GENCS) del dispositiu touch. Per la seva part, l'estructura del tipus **TSI_MemMap** defineix la mida dels registres d'E/S d'aquest dispositiu touch (tots de 32 bits) i per tant, el desplaçament (en bytes) de cadascun d'aquests registres d'E/S respecte al primer d'ells. D'aquesta forma, l'adreça del registre GENCS es la **0x40045000,** la del **DATA** es la **0x40045004** i la del **TSHD** es la **0x40045008.**
Per exemple, si des d'un programa en C es vol llegir el contingut del registre **DATA** i situar-lo en la variable sencera **reg_data**, es pot codificar de la següent forma:

```
int reg_data=TSI0_BASE_PTR->DATA;
```

Dispositius que utilitzarem en la sessió de pràctiques

Per realitzar aquesta pràctica utilitzarem els següents dispositius: els **leds** i el **touch**.

Leds

Els leds vermell i verd estan connectats als pins 18 i 19 del **PORTB** respectivament, i el led blau al pin 1 del **PORTD**. En aquesta sessió de pràctiques utilitzarem els leds vermell i verd, per tant treballarem amb els registres d'E/S del **PORTB** (**bits 18 i 19**).
Els **PORTS A, B, C, D, E**, pertanyen al mòdul **GPIO**, el qual gestiona l'E/S de propòsit general. Això vol dir que aquests PORTS tenen associats uns pins a on es poden connectar dispositius externs a la placa.
Per què pugui operar adequadament amb els leds connectats als pins del **PORTB** s'ha d'executar la següent instrucció:

```
//Activar el clock gating pels PORTB , donar energia a aquest port perquè que funcioni
SIM_BASE_PTR->SCGC5 |= SIM_SCGC5_PORTB_MASK;
```

Els registres d'E/S més importants dels PORTS del mòdul GPIO (**PORTS GPIO,** entre ells el **PORTB**) són:

- **PCR[i]:** Array de sencers a on cada element *i* d'aquest array defineix quina utilització tindrà el pin *i* del **PORT GPIO**. Per aquesta sessió de pràctiques són importants els bits 10 al 8 (MUX) de cada **PCR[i]**.
- **PDDR:** Defineix la direcció de les dades dels pins associats al **PORT GPIO** .
  - Un 1 en el PDDR[i] defineix que el pin i del **PORT GPIO** es de sortida.
  - Un 0 en el PDDR[i] defineix que el pin i del **PORT GPIO** es d'entrada.
- **PDOR:** Registre de Dades de Sortida del **PORT GPIO**
- **PCOR:** Si *PCOR[i]* val 1 s'escriu un 0 en el bit *i* del registre **PDOR**. Si *PCOR[i]* val 0 el bit *i* del registre **PDOR** no canvia.
- **PSOR:** Si *PSOR[i]* val 1 s'escriu un 1 en el bit *i* del registre **PDOR.** Si *PCOR[i]* val 0 el bit *i* del registre PDOR no canvia.

Per accedir al registre d'E/S **PCR** s'ha d'utilitzar l'apuntador **PORTB_BASE_PTR** (apuntador a l'estructura del tipus **PORT_MemMap).** Per accedir al reste de registres d'E/S s'ha d'utilitzar l'apuntador **PTB_BASE_PTR** (apuntador a l'estructura del tipus **FGPIO_MemMap)**

Touch

El dispositiu **touch** integrat en la placa està format per un parrell de elèctrodes amb capacitància. Cada elèctrode està connectat a un pin del mòdul GPIO, formant un canal. Com hi ha dos elèctrodes, existeixen dos canals: El 9, associat al pin 16 del **PORTB** i 10 associat al pin 17 d'aquest PORT.
El funcionament del **touch** implica la utilització de dos oscil·ladors: el de *capacitat*, que realitza una seqüencia de N (1, 2 fins a 32) càrregues i descàrregues del elèctrode a certa freqüència (procés conegut com a *escaneig*) i el oscil·lador de *referència,* que compta (en pulsacions, les quals dependran de la seva freqüència) quan temps triga el oscil·lador de capacitat en realitzar la seva tasca (les N carregues/descarregues). Una característica important d'aquests dos oscil·ladors, es que cadascun pot tenir la seva pròpia freqüència (no te perquè coincidir amb la de l'altre).
El dispositiu **touch** detecta el contacte (o desplaçament) d'un dit per la seva superfície perquè aquest fet provoca un canvi en el nivell de la capacitància del seu elèctrode. Aquest fet afecta a la freqüència del oscil·lador de capacitat, fent que trigui més a realitzar la seqüencia de carregues/descarregues. Aquest augment del temps es reflexa per l'oscil·lador de referència, ja que compta més pulsacions que quant no es varia la capacitància del elèctrode (i la freqüència del oscil·lador de capacitat es més gran).
A l'hora de treballar amb el **touch**, primer s'ha d'inicialitzar adequadament. Per fer això, s'han d'executar les següents instruccions:

```
//Activar el clock gating pel touch, donar energia a aquest dispositiu perquè que funcioni
SIM_SCGC5 |= (SIM_SCGC5_TSI_MASK) ;
//Configuració de les tensions i freqüències dels oscil·ladors del touch
TSI0_GENCS |= ( TSI_GENCS_MODE(0)       // Capactive sensing
              | TSI_GENCS_REFCHRG(4)     // Reference charge 4 uA
              | TSI_GENCS_DVOLT(0)       // Voltage rails
```

```
| TSI_GENCS_EXTCHRG(7)      // External osc charge
| TSI_GENCS_PS(4)           // Prescalar divide by 4
| TSI_GENCS_NSCN(11)        // Scans per electrode
| TSI_GENCS_STPE_MASK );    // Enable in STOP mode
```

Els registres d'E/S més importants del dispositiu touch són :

- **GENCS:** Registre principal de configuració del dispositiu **touch**. A part dels bits mostrats en la instrucció de configuració, també s'utilitzaran els següents bits d'aquest registre en la sessió de pràctiques:

  -TSIEN (bit 7): L'escriptura d'un 1 en aquest bit provoca l'activació del dispositiu **touch** perquè pugui operar.

  - EOSF (bit 2): Aquest bit (que actua com un flag) s'activa a 1 (sinó val 0) quan s'ha acabat el procés d'escaneig dels elèctrodes (ja s'han realitzat les N seqüencies de càrregues/descarregues). Per esborrar aquest bit s'ha d'escriure un 1 en la seva posició.

- **DATA:** Conté informació sobre el funcionament del **touch**. Els bits que s'utilitzaran en aquesta sessió de pràctiques són:

  - TSICH (bits 31 al 28) : Especifiquen el canal del **touch**.

  - TSICNT (bits 15 al 0): Conta (en pulsacions del oscil·lador de referència) el temps que tarda el elèctrode de capacitat en fer la seva tasca.

  - SWTS (bit 22): Escrivint un 1 en aquest bit, comença el procés d'escaneig dels elèctrodes.

Exercici

Implementar un programa en C que quant es passi el dit pel **touch** canvii el color del led del vermell al verd. Si no hi ha activitat en el **touch**, el color del led s'ha de mantenir en vermell.

1) Activar el *clock gating* pel **PORTB** (indicat en aquest document)
2) Inicialitzar el **touch** (les instruccions indicades en aquest document)
3) Activar el **touch**.
4) Declarar els pins 16 i 17 del port B (associats als canals del **touch**) que no seran utilitzats pel mòdul **GPIO** (ajuda: bits MUX del registre PCR).
5) Declarar els pins 18 i 19 del **PORTB** (associats al led) que seran utilitzats pel mòdul **GPIO** .
6) Declarar els pins 18 i 19 del **PORTB** com de sortida.
7) Capturar el temps que triga el **touch** a fer un escaneig inicial sense tocar la seva superfície:
   7.1) Començar un escaneig del canal 9 del **touch** .
   7.2) Esperar a què s'acabi aquest escaneig.
   7.3) Guardar el temps que ha trigat en realitzar-se aquest escaneig inicial (pista: bits TSICNT del registre DATA).
   7.4) Esborrar el bit EOSF del registre GENCS.

8) Repetir el següent procés:
   8.1) Fer els passos 7.1 i 7.2

8.2) Comprovar si el temps que s'ha trigat a fer aquest escaneig es més gran que el de l'escaneig inicial:
Si ho és, il·luminar el led amb llum verda.
Sinó continuar il·luminant el led amb llum vermella.
8.3) Esborrar el bit EOSF del registre GENCS.

Indicació important: Per accedir als registres d'E/S en aquesta sessió de pràctiques només es poden utilitzar les estructures: ***DISPOSITIU_MemMap i DISPOSITIU_BASE_PTR.*** No es poden utilitzar les macros (ni les mascares) que ofereix el fitxer **MKL25Z4.h** per accedir als diferents registres d'E/S definits en DISPOSITIU_MemMap.

Documentació

- Al Campus Virtual: Dispositius i registres d'E/S de placa FRDM-KL25Z: *FRDM-KL25Z User's Manual i KL25 Sub-Family Reference Manual.*
- Operadors binaris amb C: http://www.zator.com/Cpp/E4_9_3.htm