advanced digital chips inc

Preliminary

CANTUS

- *Key Scan* -

32bits EISC Microprocessor *CANTUS*

Ver 1.0
October 8, 2009

Advanced Digital Chips Inc.

History

2009-10-08	Created Preliminary Specification

CANTUS Evaluation Board Application Note : #0004 Key Scan

Office

8th Floor, KookMin 1 Bldg.,

1009-5, Daechi-Dong, Gangnam-Gu, Seoul, 135-280, Korea

Tel : + 82-2-2107-5800

Fax : + 82-2-571-4890

URL : http://www.adc.co.kr

— Table of Contents —

— List of Figures —

— List of Tables —

1 Summary

이 문서는 CANTUS SDK의 Key Scan에 대한 Application Note이다.

CANTUS에는 최대 4 x 4의 Key Matrix를 제어할 수 있는 Key Scan Controller가 내장되어 있다.

SDK 예제의 Key Scan Project는 Key Scan Controller를 사용하여 6개의 Switch로 구성된 Key Matrix를 Key Press / Release Mode로 Key를 식별한다. Key의 Press, Release일 때 Interrupt를 발생하여 ISR에서 Buffer에 저장하고, main() 에서 getkeyscan() 함수를 호출하여 ISR에서 Buffer에 저장한 Key값을 읽어 상태를 UART로 출력한다.

- Key Scan Controller에 대한 자세한 내용은 CANTUS Datasheet 18 KEYSCAN을 참조하라.
- UART는 AN_0002_UART를 참조하라.
- Interrupt는 AN_0003_INTERRUPT를 참조하라.
- Timer는 AN_0005_TIMER를 참조하라.

2 Register Set

2.1 Register Set Flow Chart

Key Scan Controller를 사용하기 위해선 다음과 같은 순서로 Register를 설정한다.

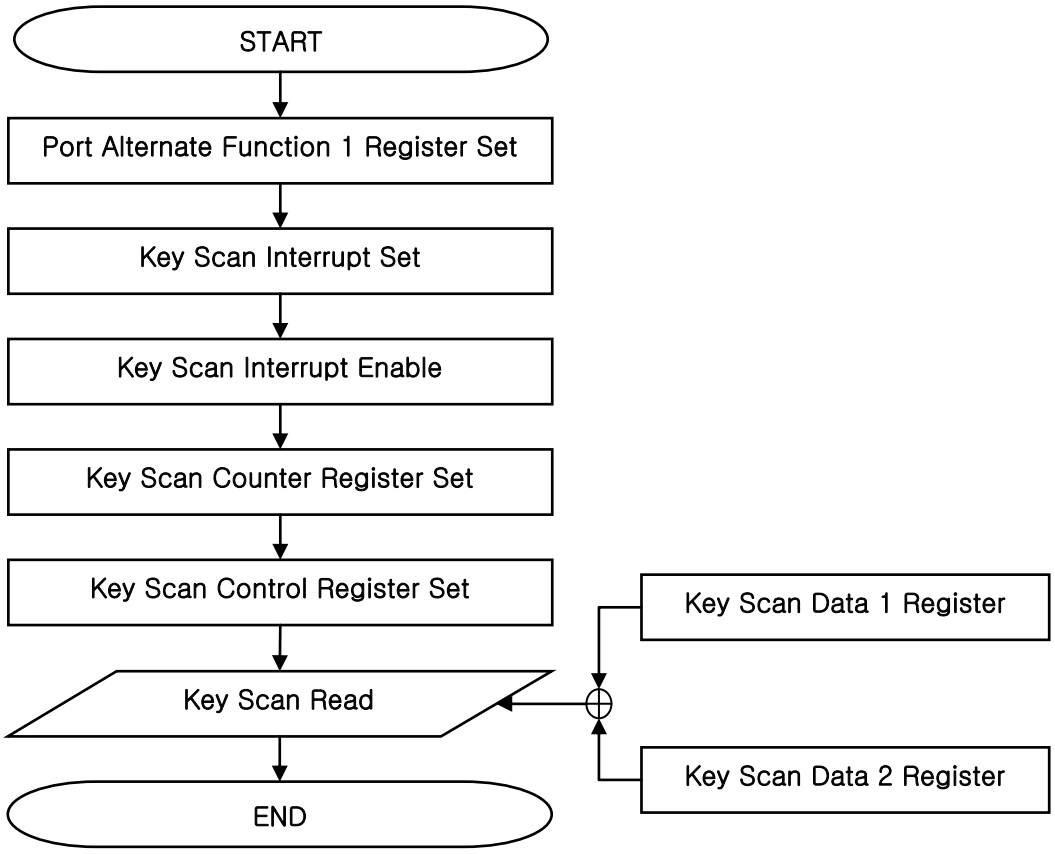

그림 2-1 Register Set Flow Chart

2.2 Port Alternate Function Register

Key Scan Controller는 PIN 2~9를 사용한다. 이들 PIN은 아래 표 2-1과 같이 다른 Function을 공유 하고 있어 Key Scan Controller를 사용하기 위해서는 Port Alternate Function 1 Register에서 ***2nd*** 로 설정하여야 한다.[1]

Key Scan Project는 PIN2, 3, 4, 5, 6을 KEY_O[0], KEY_I[0], KEY_O[1], KEY_I[1], KEY_O[2]로 설정한다.[2]

표 2-1 Port 1 Alternate Function

Group	***Index***	***Pin***	***1st***	***2nd***	***3rd***	***4th(default)***
PAF1	**0**	2	TX[4]	KEY_O[0]	AD[8]	P1.0
	1	3	RX[4]	KEY_I[0]	AD[9]	P1.1
	2	4	TX[5]	KEY_O[1]	AD[10]	P1.2
	3	5	RX[5]	KEY_I[1]	AD[11]	P1.3
	4	6	TX[6]	KEY_O[2]	AD[12]	P1.4
	5	7	RX[6]	KEY_I[2]	AD[13]	P1.5
	6	8	TX[7]	KEY_O[3]	AD[14]	P1.6
	7	9	RX[7]	KEY_I[3]	AD[15]	P1.7

표 2-2 Port Alternate Function 1 Register (PAF1)

Address : 0x8002_0024

Bit	***R/W***	***Description***	***Default Value***
31 : 16	R	Reserved	-
15 : 14	R/W	P1.7 : P1.7 Port Selection bit 00 : RX[7] 01 : KEYI[3] 10 : AD[15] 11 : P1.7	11
13 : 12	R/W	P1.6 : P1.6 Port Selection bit 00 : TX[7] 01 : KEYO[3] 10 : AD[14] 11 : P1.6	11
11 : 10	R/W	P1.5 : P1.5 Port Selection bit 00 : RX[6] 01 : KEYI[2] 10 : AD[13] 11 : P1.5	11
9 : 8	R/W	P1.4 : P1.4 Port Selection bit 00 : TX[6] 01 : KEYO[2] 10 : AD[12] 11 : P1.4	11
7 : 6	R/W	P1.3 : P1.3 Port Selection bit 00 : RX[5] 01 : KEYI[1] 10 : AD[11] 11 : P1.3	11
5 : 4	R/W	P1.2 : P1.2 Port Selection bit 00 : TX[5] 01 : KEYO[1] 10 : AD[10] 11 : P1.2	11
3 : 2	R/W	P1.1 : P1.1 Port Selection bit 00 : RX[4] 01 : KEYI[0] 10 : AD[9] 11 : P1.1	11
1 : 0	R/W	P1.0 : P1.0 Port Selection bit 00 : TX[4] 01 : KEYO[0] 10 : AD[8] 11 : P1.0	11

[1] CANTUS Datasheet의 8 GPIO 참조.

[2] Evaluation Board를 위한 기타 Port Alternate Function은 회로와 boardinit.c를 참조하라.

2.3 Key Scan Interrupt Set

Key Scan Controller는 Key가 입력되면 Interrupt를 발생시킨다. 이 Interrupt를 사용하기 위해 interrupt.c 의

```
BOOL setinterrupt(INTERRUPT_TYPE intnum,void (*fp)());
```

를 호출하여 KEYSCAN Interrupt Vector Number 0x3D에 ISR로 사용할 KeyScan_ISR을 Setting 한다.

2.4 Key Scan Interrupt Enable

```
void EnableInterrupt(INTERRUPT_TYPE num,BOOL b);
```

Interrupt Service Routine(ISR)이 Vector Table에 Setting되면 Interrupt를 Enable 한다.

이상 위 **2.3**과 **2.4**에 해당하는 Source부분은 아래와 같다.

```
setinterrupt(INTNUM_TIMER7_KEYSCAN, KeyScan_ISR);
EnableInterrupt(INTNUM_TIMER7_KEYSCAN, TRUE);
```

2.5 Key Scan Counter Register

Key Scan Controller는 일정 주기로 Key를 Scan한다. Key Scan Counter Register는 Scanning Frequency를 설정하는 상수이며, 아래와 같이 Scanning Frequency가 결정된다.

표 2-3 Key Scan Counter Register(KSCNT)

Address : 0x8002_3004

Bit	*R/W*	*Description*	*Default Value*
31 : 16	R	Reserved	-
15 : 0	R/W	Scan clock divide ratio setting bits.	0xFFFF

$$Scanning\ Frequency = \frac{f_{PCLK}}{11 \times (KSCNT + 1)}$$

2.6 Key Scan Control Register

Key Scan Control Register는 Key Scan Controller의 동작 Mode를 변경하고, 동작을 On/Off 할 수 있다.

표 2-4 Key Scan Control Register(KSCTRL)

Address : 0x8002_3000

Bit	*R/W*	*Description*	*Default Value*
31 : 3	R	Reserved	-
2	R/W	Key Scan Mode Select bits 0 : Key press mode 1 : Key press / release mode	0
1	R	Reserved	-
0	R/W	Key Scan Enable bit 0: Scan Disable 1: Scan Enable	0

2.7 Key Scan Data 1 Register

Key Scan의 결과로 식별된 Key의 값은 Key Scan Data 1 Register를 통해 알 수 있다.
Key Scan Data 1 Register는 입력에 해당하는 Bit가 Set 된다.

표 2-5 Key Scan Data 1 Register(KSD1)

Address: 0x8002_3008

Bit	*R/W*	*Description*	*Default Value*
31 : 16	R	Reserved	-
15	R	SW33	0
14	R	SW32	0
13	R	SW31	0
12	R	SW30	0
11	R	SW23	0
10	R	SW22	0
9	R	SW21	0
8	R	SW20	0
7	R	SW13	0
6	R	SW12	0
5	R	SW11	0
4	R	SW10	0
3	R	SW03	0
2	R	SW02	0
1	R	SW01	0
0	R	SW00	0

2.8 Key Scan Data 2 Register

Key Scan의 결과로 식별된 Key의 값은 Key Scan Data 2 Register를 통해 알 수 있다.
Key Scan Data 2 Register는 입력된 Key를 Hexadecimal로 표현한다.

표 2-6 Key Scan Data 2 Register(KSD2)

Address: 0x8002_300C

Bit	*R/W*	*Description*	*Default Value*
31:5	R	Reserved	-
4: 0	R	Scanned Switch value (Represented as Hexadecimal Value)	0x00

*** Key Scan Data 1 Register의 값을 십육진수로 표현한다. (SW03는 0x04로 표현)
*** (주의) 2개 이상의 Key가 눌러지면 이 Register의 값은 "0"으로 Setting 된다

이상 다음 입력이 식별 되면 KSD1과 KSD2 Register를 읽었을 때 다음과 같은 값을 얻을 수 있다.

Ex) SW00가 입력으로 식별되면 KSD1 : 0x01, KSD2 : 0x01
SW20가 입력으로 식별되면 KSD1 : 0x100, KSD2 : 0x09

3 Function Set

3.1 Function Set Flow Chart

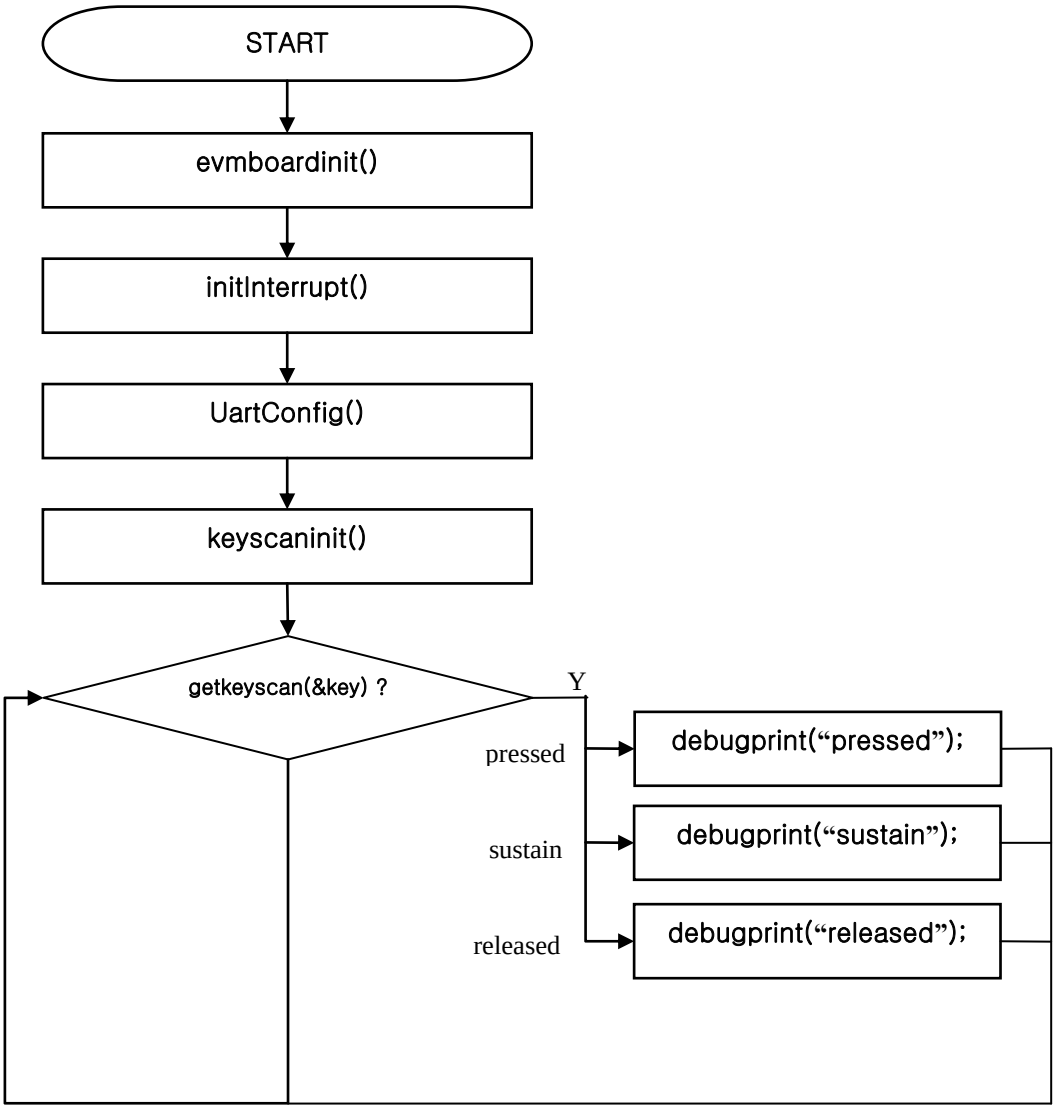

그림 3-1 Function Set Flow Chart

3.1.1 keyscaninit()

void keyscaninit(KEYSCAN_MODE mode);

- KEYSCAN_MODE mode : KEYMODE_PRESS / KEYMODE_BOTH

```
void keyscaninit(KEYSCAN_MODE mode)
{
        *R_PAF1=(*R_PAF1&~0x3ff)|
          PAF1_KEYO0|PAF1_KEYI0|PAF1_KEYO1|PAF1_KEYI1|PAF1_KEYO2 ;
        *R_KSCNT = 0xffff;//default
#ifdef KEY_SUSTAIN_SUPPORT
        if(mode == KEYMODE_BOTH)
        {
                INTERRUPT_TYPE intnum = SUSTAIN_TIMER*4+1;
                setinterrupt(intnum,suschecktimerisr);
        }
#endif
        KeyScanBufClear();
        setinterrupt(INTNUM_TIMER7_KEYSCAN,KeyScan_ISR);
        EnableInterrupt(INTNUM_TIMER7_KEYSCAN,TRUE);
        *R_KSCTRL = mode|KEYCON_ENABLE;
}
```

Key Scan Controller를 사용하기 위한 초기화 과정으로 PORT Alternate Function과 Key Scan Controller의 Register Setting, setinterrupt()와 Enableinterrupt() 함수를 호출한다.

3.1.2 getkeyscan()

BOOL getkeyscan(KEY* key);

- Return : Key buffer에 저장된 값이 있으면 True
- KEY* key : key를 저장할 구조체

```
BOOL getkeyscan(KEY* key)
{
#ifdef KEY_SUSTAIN_SUPPORT
        if(*R_KSCTRL & KEYCON_BOTH_MODE)
        {
                CRITICAL_ENTER();
                if(bsustain)
                {
                        if(oldkeydata)
                        {
                                key->type = KEYTYPE_SUSTAIN;
                                key->val = oldkeydata;
                                CRITICAL_EXIT();
                                return TRUE;
                        }
                        bsustain = FALSE;
                }
        }
#endif
        CRITICAL_ENTER();
        if(keybuf.Ctr>0)
        {
                key->type = keybuf.OutPtr->type;
                key->val = keybuf.OutPtr->val;
                keybuf.OutPtr++;
                if(keybuf.OutPtr == &keybuf.key[KEY_BUF_SIZE])
                {
                        keybuf.OutPtr = &keybuf.key[0];
                }
                keybuf.Ctr--;
                CRITICAL_EXIT();
                return TRUE;
        }
        CRITICAL_EXIT();
        return FALSE;
}
```

KeyScan_ISR()에서 Key Buffer에 저장된 Key값을 차례로 읽어 온다.

3.2 keyscan.c

/Cantuslib/keyscan.c

keyscan.c는 key Scan Controller를 사용하기 위한 정의 및 함수로 구성되어 있다. 사용자는 Key Scan Controller의 Register를 직접 사용하지 않아도 keyscan.c에서 정의된 함수를 사용하여 Key Scan Controller를 설정하여 사용할 수 있다.

Key의 눌려진 시간을 판별하기 위해 Timer 6번을 사용하여 일정 시간동안 Key가 눌려지면 이를 연속적인 입력으로 판단한다.

4 Point This Note

- Key Scan Controller는 Key Press, Key Release에서 Interrupt를 발생한다.
- Key Scan Data 1 Register와 Key Scan Data 2 Register를 통해 Scan한 결과를 알 수 있다.
- 사용자가 Key를 확장하여 설계하고자 할 경우 아래 그림과 같이 설계하여야 한다.
- Evaluation Board의 Switch는 SW00, SW01, SW10, SW11, SW20, SW21이 구성되어 있다.

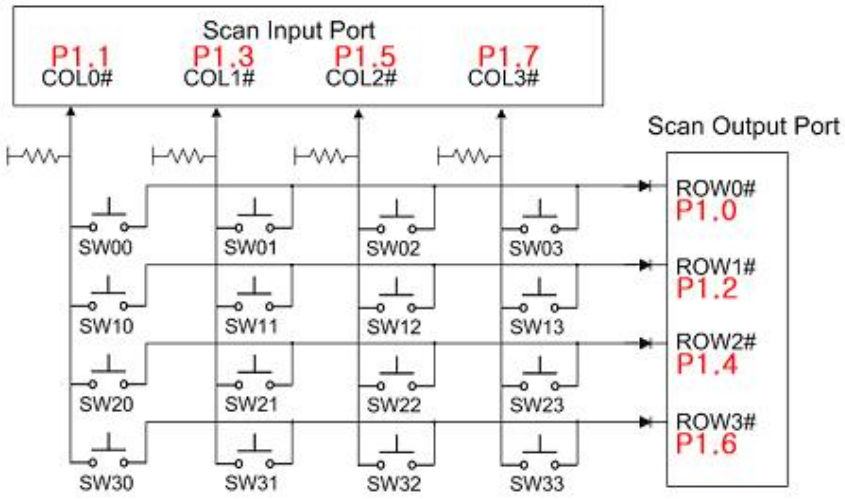

그림 4-1 4 x 4 Key Matrix

Evaluation Board Schematic에서 나타내는 각 Switch 넘버는 아래 표와 같이 Key Matrix에 대응된다.

표 4-1 Evaluation Board key

Schematic	*Key Matrix*	*KSD1*	*KSD2*
SW3	SW20	0x100	0x9
SW4	SW10	0x10	0x5
SW5	SW00	0x1	0x1
SW6	SW21	0x200	0xa
SW7	SW11	0x20	0x6
SW8	SW10	0x2	0x2