MC43. Médian printemps 2013.

Sans documents. Sans calculatrice.

Capteur de température SPI.

Le capteur de température ADT7310 est un capteur de température équipé d'une interface de communication SPI. Il est piloté par un contrôleur DSP TMS320LF2403 cadencé à 40 MHz. Dans cet exercice, on ne traite que la configuration du bus SPI et la lecture de la température. Les lectures de température se font à vitesse maximale par paquets de données de 16 bits.

A.1) En vous aidant des datasheets, **donner** le schéma de câblage entre le DSP et le capteur de température.

A.2) D'après le timing de lecture, **indiquer** comment configurer les bits Clock Polarity et Clock Phase des registres SPICCR et SPICTL afin d'adapter les formes des signaux au capteur de température.

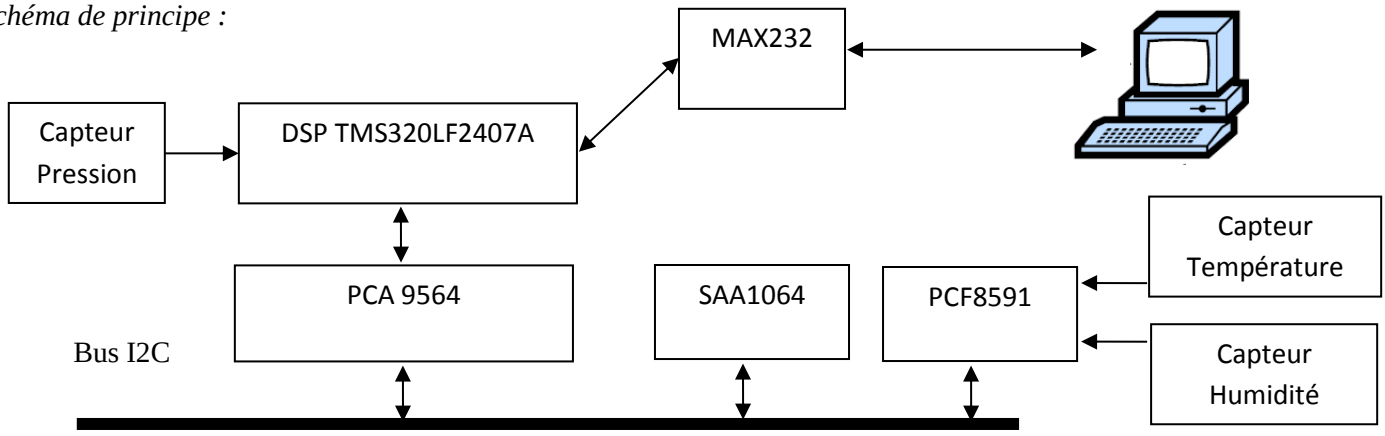
A.3) **Donner** le contenu des registres d'initialisation du contrôleur SPI du DSP : SPICCR, SPIBRR et SPICTL pour que les transferts soient cadencés à la fréquence SPI maximale possible.

A.4) **Ecrire** en langage C le sous-programme ***int temperature(void)*** permettant de réaliser une mesure de température.

Réalisation d'un capteur de pression.

On souhaite réaliser un capteur de pression à partir d'un DSP contrôleur TMS320LF2407A cadencé à 40MHz. Outre la pression, ce capteur doit mesurer la température et l'humidité. Il doit afficher la valeur de la pression corrigée et la communiquer à un PC via une liaison série RS232.

Schéma de principe :



A chaque mesure de pression est associée une mesure de température et une mesure d'humidité. Lorsque les trois mesures sont effectuées, le DSP réalise un traitement (non étudié ici) et transmet le résultat de la mesure de pression au PC via la liaison RS232 ainsi qu'aux deux afficheurs 7 segments pilotés par le SAA1064.

L'accès aux registres des composants utilisés s'effectue en utilisant le nom du datasheet. (Ex I2CCON, ADCTRL1 …)

Partie 1 : **Mesure de la pression.**

Le capteur de pression est relié à la voie 1 du convertisseur analogique numérique du DSP. Il délivre une tension comprise entre 0 et 3,3V. La fréquence de la mesure est de 2kHz, la conversion est déclenchée par le Timer3 du DSP. La lecture des résultats de conversion s'effectue par la fonction d'interruption liée à INT1 et déclenchée par le module ADC.

1.1) **Ecrire** en langage C la fonction **void INITTIMER(void)** initialisant le timer3 pour cadencer l'échantillonnage et déclencher les conversions analogiques numériques.
1.2) **Ecrire** en langage C la fonction **void INITADC(void)** initialisant le convertisseur analogique numérique pour permettre son fonctionnement en mode cascadé et start stop.
1.3) **Indiquer** (en assembleur) comment initialiser les vecteurs d'interruption pour associer l'interruption INT1 du DSP à la fonction : ***interrupt void interADC(void)***.
1.4) **Ecrire** en langage C la fonction **interrupt void inter(void)** qui récupère le résultat de la conversion analogique numérique dans la variable pression.

Partie 2 : **Mesure température et humidité.**

Le capteur de température est relié à l'entrée AIN0 du PCF8591 et le capteur d'humidité est relié à l'entrée AIN1 du même composant. Ils délivrent chacun une tension comprise entre 0 et 5V. L'adresse du PCF8591 est 93h en lecture et 92h en écriture. Le DSP accède au bus I2C par l'intermédiaire du contrôleur I2C PCA9564.

2.1) **Donner** le câblage des broches A2, A1 et A0 du PCF8591.
2.2) **Indiquer** comment agir sur les registres du PCA9564 pour envoyer une condition de START.
2.3) **Indiquer** comment agir sur les registres du PCA9564 pour envoyer une condition de STOP.
2.4) **Indiquer** comment configurer le contrôleur PCA9564 pour qu'il effectue un acquittement après réception d'une donnée.
2.5) **Donner**, en le justifiant, l'octet de configuration du PCF8591.
2.6) **Donner** la trame I2C permettant de lire la température et la pression.

On rappelle quelques macros et fonctions utiles :

```
#define BIT(n,x)        ((1<<(n))&(x))          /* Teste le bit n de x */
#define SETBIT(n,x)     ((x) |= (1<<(n)))       /* Met a 1 le bit n de x */
#define CLRBIT(n,x)     ((x) &= ~(1<<(n)))      /* Met a 0 le bit n de x */
#define INVBIT(n,x)     ((x) ^= (1<<(n)))       /* Inverse le bit n de x */

/* Appellation des bits des registres du contrôleur PCA9564 */
#define SI      3
#define STO     4
#define STA     5
#define AA      7
```

- **Initialisation du contrôleur I2C PCA9564**

```
void init_PCA9564(void)
{
  MCRB &= 0xFFFD;                                  /* active IOPC1 */
  PCDATDIR = (PCDATDIR | 0x0200) & 0xFFFD;         /* Reset du contrôleur I2C connecté sur IOPC1 */
  PCDATDIR |= 0x0002;                              /* désactive le reset du contrôleur I2C */
  I2CCON = 0x0045;                                 /* active le contrôleur, F = 59 kHz */
}
```

2.7) **Ecrire** en langage C les routines suivantes : config_PCF (configuration du convertisseur A/N), mesure_Temperature (lecture de la conversion A/N de la température) et mesure_Humidite (lecture de la conversion A/N de l'humidité).

Partie 3 : **Affichage de la pression.**

La pression est affichée dans le mode suivant :

- Adresse 0x71
- 2 digits continus : un chiffre pour les dizaines (variable D) et un chiffre pour les unités (variable U)
- Courant de commande des afficheurs 15mA.

3.1) **Donner** la séquence d'octets à envoyer sur le bus I2C. On distinguera la première séquence des suivantes. Vous préciserez la valeur des octets de la séquence I2C.
3.2) **Indiquer** comment est câblé le circuit pour l'adresse 0x71.

Partie 4 : **Liaison RS232.**

La liaison série doit être configurée de la manière suivante : 19200 bauds, données de 8 bits, 1 bit de parité (parité paire) et 1 bit de stop.
La donnée à envoyer (mesure de pression après correction due à l'humidité et à la température) est codée sur 16 bits. Le poids faible est toujours envoyé en premier.

4.1) **Représenter** la trame RS232 (signal SCITXD) avant et après le circuit MAX232 pour l'envoi de l'octet 0xA1.
4.2) **Ecrire** en langage C le sous-programme ***init_SCI*** configurant la liaison série du DSP.
4.3) **Ecrire** en langage C la fonction d'émission d'un octet ***void putchar(unsigned int a)***.
4.4) **Déterminer** la vitesse de transmission utile maximale (bande passante) en octets par seconde.

Documentation capteur de température ADT7310 :

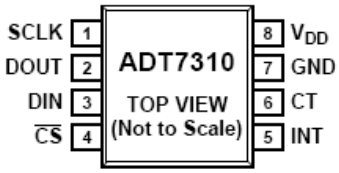


Figure 5. Pin Configuration

Table 4. Pin Function Descriptions

Pin No.	Mnemonic	Description
1	SCLK	Serial Clock Input. The serial clock is used to clock in and clock out data to and from any register of the ADT7310.
2	DOUT	Serial Data Output. Data is clocked out on the SCLK falling edge and is valid on the SCLK rising edge.
3	DIN	Serial Data Input. Serial data to be loaded to the part's control registers is provided on this input. Data is clocked into the registers on the rising edge of SCLK.
4	$\overline{CS}$	Chip Select Input. The device is selected when this input is low. The device is disabled when this pin is high.
5	INT	Overtemperature and Undertemperature Indicator. Logic output. Power-up default setting is as an active low comparator interrupt. Open-drain configuration. A pull-up resistor is required, typically 10 kΩ.
6	CT	Critical Overtemperature Indicator. Logic output. Power-up default polarity is active low. Open-drain configuration. A pull-up resistor is required, typically 10 kΩ.
7	GND	Analog and Digital Ground.
8	V_{DD}	Positive Supply Voltage (2.7 V to 5.5 V). The supply should be decoupled with a 0.1 µF ceramic capacitor to ground.

Timing de lecture d'un registre 16 bits :

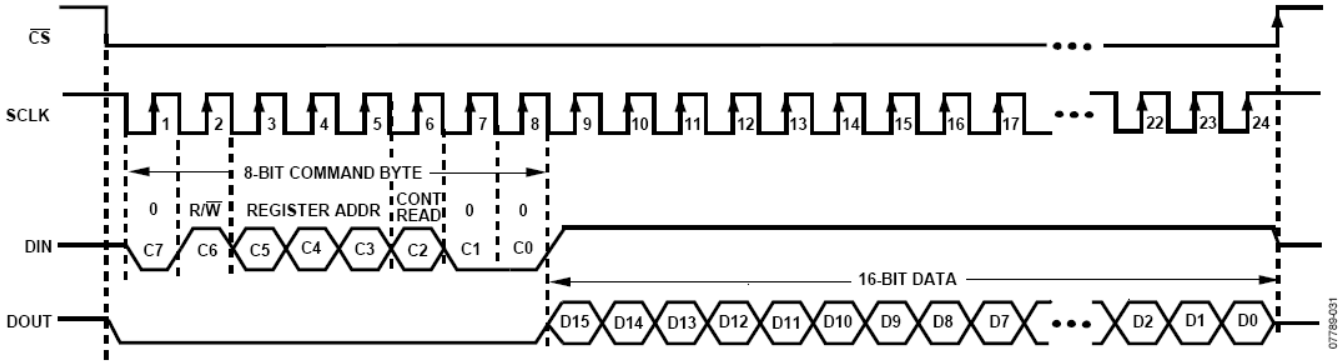


Figure 21. Read from a 16-Bit Register

Lecture en mode continu :

CONTINUOUS READ MODE

When the command byte = 01010100 (0x54), the contents of the temperature value register can be read out without requiring repeated writes to the communications register. By sending 16 SCLK clocks to the ADT7310, the contents of the temperature value register are output onto the DOUT pin.

To exit the continuous read mode, the Command Byte 01010000 (0x50) must be written to the ADT7310.

While in continuous read mode, the part monitors activity on the DIN line so that it can receive the instruction to exit the continuous read mode. Additionally, a reset occurs if 32 consecutive 1s are seen on the DIN pin. Therefore, hold DIN low in continuous read mode until an instruction is to be written to the device.

In continuous read mode, the temperature value register cannot be read when a conversion is taking place. If an attempt is made to read the temperature value register while a conversion is taking place, then all 0s are read. This is because the continuous read mode blocks read access to temperature value register during a conversion.

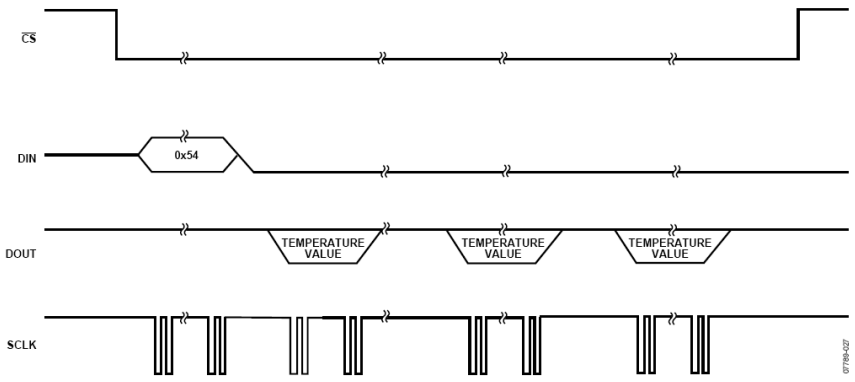


Figure 16. Continuous Read Mode