MC60

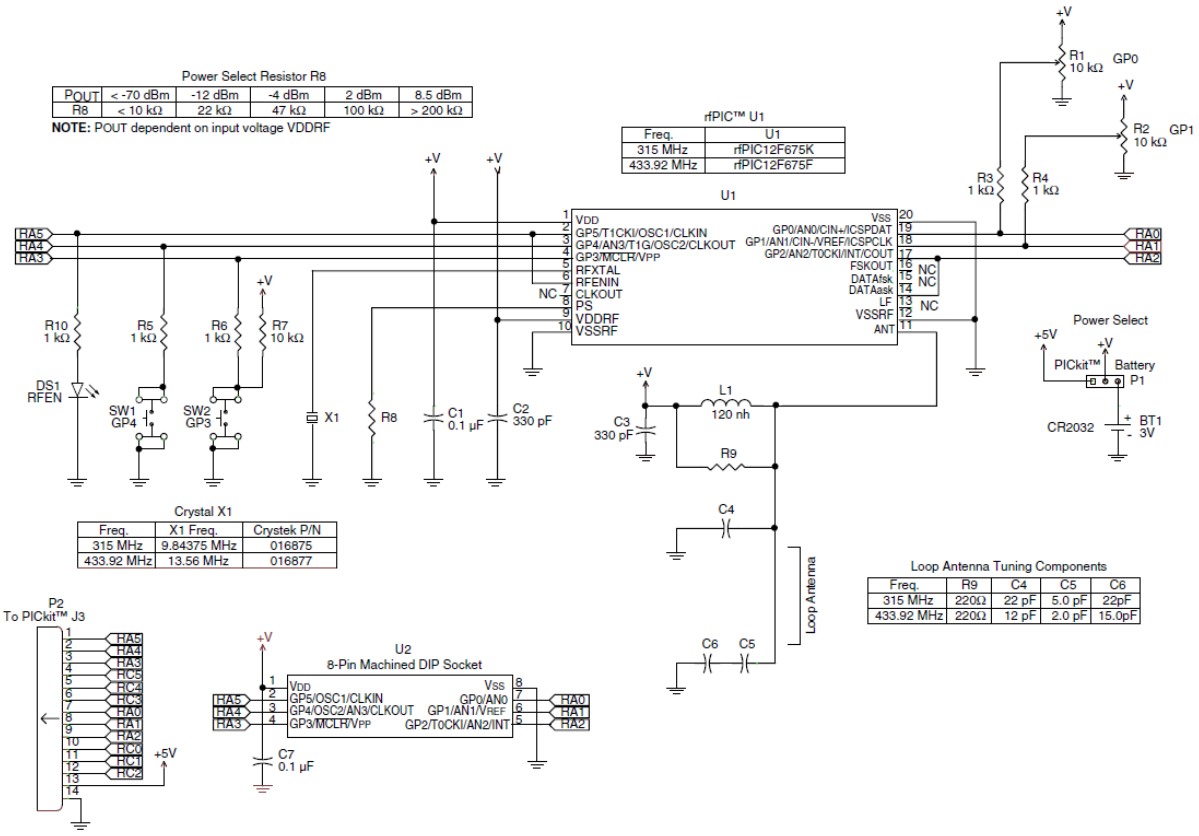
Examen Final

Automne 2015

Télécommande pour porte de garage

Présentation

Le microcontrôleur rfPIC12F675 possède un émetteur radio intégré, auquel il ne reste qu'à ajouter quelques composants de filtrage et d'adaptation. Le schéma illustre la simplicité de mise en œuvre.



Il s'agit d'un schéma provenant d'un kit d'évaluation à destination de bureaux d'études. Il comporte des composants supplémentaires pour l'étude de différentes fonctionnalités et la possibilité de les reprogrammer à volonté. Ces composants, qui ne sont pas concernés par l'étude, et qui peuvent donc être supprimés sont :

- Les potentiomètres R1 et R2, ainsi que les résistances associées R3 et R4, utilisés pour tester le convertisseur Analogique/Numérique
- Le socle U2, destiné à recevoir un autre microcontrôleur
- Le connecteur P2, qui permet de brancher cette télécommande sur un circuit de programmation
- Le connecteur P1 qui permet de sélectionner l'alimentation (par pile ou par le connecteur P2 s'il est connecté)

Les composants qui interviennent dans cette étude sont la LED rouge (V_{seuil}=1.4V) DS1 qui indique le passage en mode émission, ainsi que les boutons poussoirs SW1 et SW2. Les autres composants

(condensateurs, quartz, inductance) sont indispensables au fonctionnement et imposés par le fabricant (indications figurant sur le Datasheet du PIC).

Précisons que la résistance R7 (Pull-Up) est facultative et pourra être considérée comme absente (une configuration appropriée des ports d'entrée-sortie GPx permet de s'en affranchir).

Schéma

Question 1

Cette télécommande ne possède que deux boutons. On souhaite ajouter deux boutons supplémentaires pour pouvoir commander plus de périphériques. Redessinez une partie du schéma représentant la connexion des boutons SW3 et SW4.

Question 2

La LED DS1 s'allume lorsque le Port GP5 passe à 1 (alimentation par pile lithium 3.2V). Quelle est alors l'intensité du courant traversant la LED ?

Logiciel

L'horloge de cadencement est fixée à 4 MHz

Question 3 - Fonction de délai de 400µs x W

Les données sont émises bit par bit en mode série. Pour cela, on a besoin d'une fonction dont la durée est d'environ 400 µs fois le contenu du registre W. Dans le programme de démonstration fourni, cette fonction s'appelle WaitxTE. Elle utilise des boucles d'instructions pour réaliser le délai souhaité. On vous propose de vérifier dans un premier temps la durée réelle de cette fonction. Le tableau fourni en annexe décrit les instructions, et donne leur durée d'exécution.

`Count2` et `Count` sont des registres généraux de 1 octet définis au début du programme.

```
;--------------------------------------------------------------------
; Subroutine: WaitxTE
; Initialization: W x 400us length of delay required
; Output: Count, Count2   as decrementing counters
;--------------------------------------------------------------------
WaitxTE
        movwf   Count2
waitxlp
        movlw   D'79'    ; idem movlw .79
        movwf   Count
wait400lp
        nop
        nop
        decfsz  Count,F
        goto    wait400lp
;
        decfsz  Count2,F
        goto    waitxlp
        retlw   0
```

Question 3a – Début de temporisation

Conseil : Pour calculer la durée d'une boucle, essayer avec des contenus de compteur particuliers (1, 2, 3) et détaillez vos calculs.

Que contient le registre Count lorsque l'on atteint pour la première fois l'étiquette `waitxlp` ?

Question 3b – Boucle centrale

Combien de temps faut-il pour, partant de l'étiquette `wait400lp`, atteindre l'instruction `decfsz Count2,F` ?

Question 3c – Boucle complète

Etablir la relation exacte donnant la durée de la fonction `WaitxTE` en fonction du contenu du registre W

Question 3d – Cas particulier

Quelle est la durée de la fonction si W contient 0 lors de l'appel ?

Question 3e – Fin de fonction

Que contient le registre W au retour de la fonction ?

Programmation en C

Question 4 – Simple programme d'essai

Compléter le programme suivant pour que la LED s'allume si au moins un des deux boutons est pressé, et reste allumée tant qu'il est pressé. Ce programme est incomplet, les initialisations sont omises.

```
#define BP1 GPIO4
#define BP2 GPIO3
#define LED GPIO5
#define PRESSE 0
#define LACHE 1
…
//déclarer ici vos variables si nécessaire

void main(void)
{
// Les initialisations des Ports et autres ne sont pas demandées.

LED = 0;

// Placer vos instructions ici

}
```

Annexe

Jeu d'instructions disponibles

rfPIC12F675

TABLE 11-2: rfPIC12F675 INSTRUCTION SET

Mnemonic, Operands		Description	Cycles	14-Bit Opcode MSb LSb	Status Affected	Notes
BYTE-ORIENTED FILE REGISTER OPERATIONS						
ADDWF	f, d	Add W and f	1	00 0111 dfff ffff	C,DC,Z	1,2
ANDWF	f, d	AND W with f	1	00 0101 dfff ffff	Z	1,2
CLRF	f	Clear f	1	00 0001 lfff ffff	Z	2
CLRW	-	Clear W	1	00 0001 0xxx xxxx	Z	
COMF	f, d	Complement f	1	00 1001 dfff ffff	Z	1,2
DECF	f, d	Decrement f	1	00 0011 dfff ffff	Z	1,2
DECFSZ	f, d	Decrement f, Skip if 0	1(2)	00 1011 dfff ffff		1,2,3
INCF	f, d	Increment f	1	00 1010 dfff ffff	Z	1,2
INCFSZ	f, d	Increment f, Skip if 0	1(2)	00 1111 dfff ffff		1,2,3
IORWF	f, d	Inclusive OR W with f	1	00 0100 dfff ffff	Z	1,2
MOVF	f, d	Move f	1	00 1000 dfff ffff	Z	1,2
MOVWF	f	Move W to f	1	00 0000 lfff ffff		
NOP	-	No Operation	1	00 0000 0xx0 0000		
RLF	f, d	Rotate Left f through Carry	1	00 1101 dfff ffff	C	1,2
RRF	f, d	Rotate Right f through Carry	1	00 1100 dfff ffff	C	1,2
SUBWF	f, d	Subtract W from f	1	00 0010 dfff ffff	C,DC,Z	1,2
SWAPF	f, d	Swap nibbles in f	1	00 1110 dfff ffff		1,2
XORWF	f, d	Exclusive OR W with f	1	00 0110 dfff ffff	Z	1,2
BIT-ORIENTED FILE REGISTER OPERATIONS						
BCF	f, b	Bit Clear f	1	01 00bb bfff ffff		1,2
BSF	f, b	Bit Set f	1	01 01bb bfff ffff		1,2
BTFSC	f, b	Bit Test f, Skip if Clear	1 (2)	01 10bb bfff ffff		3
BTFSS	f, b	Bit Test f, Skip if Set	1 (2)	01 11bb bfff ffff		3
LITERAL AND CONTROL OPERATIONS						
ADDLW	k	Add literal and W	1	11 111x kkkk kkkk	C,DC,Z	
ANDLW	k	AND literal with W	1	11 1001 kkkk kkkk	Z	
CALL	k	Call subroutine	2	10 0kkk kkkk kkkk		
CLRWDT	-	Clear Watchdog Timer	1	00 0000 0110 0100	$\overline{TO}$,$\overline{PD}$	
GOTO	k	Go to address	2	10 1kkk kkkk kkkk		
IORLW	k	Inclusive OR literal with W	1	11 1000 kkkk kkkk	Z	
MOVLW	k	Move literal to W	1	11 00xx kkkk kkkk		
RETFIE	-	Return from interrupt	2	00 0000 0000 1001		
RETLW	k	Return with literal in W	2	11 01xx kkkk kkkk		
RETURN	-	Return from Subroutine	2	00 0000 0000 1000		
SLEEP	-	Go into Standby mode	1	00 0000 0110 0011	$\overline{TO}$,$\overline{PD}$	
SUBLW	k	Subtract W from literal	1	11 110x kkkk kkkk	C,DC,Z	
XORLW	k	Exclusive OR literal with W	1	11 1010 kkkk kkkk	Z	

Note 1: When an I/O register is modified as a function of itself (e.g., `MOVF GPIO, 1`), the value used will be that value present on the pins themselves. For example, if the data latch is '1' for a pin configured as input and is driven low by an external device, the data will be written back with a '0'.

2: If this instruction is executed on the TMR0 register (and, where applicable, d = 1), the prescaler will be cleared if assigned to the Timer0 module.

3: If Program Counter (PC) is modified, or a conditional test is true, the instruction requires two cycles. The second cycle is executed as a NOP.

Attention : 1 cycle d'instruction nécessite 4 périodes d'horloge de cadencement.