Premiers pas avec MPLAB 8.0

Microchip propose pour le développement de ses microcontrôleurs, un « Environnement de Développement Intégré » ou IDE, sous la forme d'un logiciel du nom de MPLAB, téléchargeable gratuitement sur le site de Microchip.

MPLAB permet, entre autres, d'éditer le code source de votre programme en assembleur, de le compiler, le simuler, de déboguer à l'aide du module ICD2 par exemple, et de programmer le circuit cible à l'aide ce même module.

1 Créer un projet

Après avoir lancé le programme par « **Démarrer -> Tous les Programmes -** > **Microchip MPLAB IDE** »,

créer un nouveau projet à l'aide de l'assistant :

MPLAB IDE v8.00 - Untitled Workspace	E F	Project Wizard		
File Edit View Project Debugger Programmer Tools C	I	Step One: Select a device		
Project Wizard New Open Close Set Active Project			Device: Pterise3772 CPrécédent Suivant> Annuler Aide	

Choisir le circuit cible (PIC16F877 ou 16F877A) lors de la première phase, puis les outils lors de la seconde (laisser les options par défaut) et le répertoire de travail lors de la troisième phase. On pourra créer par exemple un répertoire TP1.

	AVE FLUIEULAS	
Project Wizard Step Two: Select a language toolsuite	Enregister dans : Prog pic	Créer un nouveau dossier
Active Toolsuite: Microchip MPASM Toolsuite Active Toolsuite: Microchip MPASM Toolsuite MPASM Assembler (mplink.exe) MPLIN Diject Linker (mplink.exe) MPLIN Diject Linker (mplink.exe) Location C:\Program Files\Microchip\MPASM Suite\MPAsmWin.exe Browse HelpI My Suite Isn't Listed! Show all installed toolsuites (Précédent Suivant> Annuler Aide	BED TEMPO 18F252 Project Wizard Step Three: Create a new project, or reconfigure the active project? Create New Project File D:\prog pic\TP1_mplab\TP1 Reconfigure Active Project Make changes without saving Save changes to another project file Save changes to another project file	Créer un nouveau dossier

Ajouter ensuite le fichier « PROG_1.asm » qui se trouve dans le répertoire « Ressources / T[•]P1 ». Ce programme est celui étudié au chapitre précédent et est reproduit en annexe 1. La dernière fenêtre récapitule les actions précédentes.

Project Wizard	Project Wizard
Step Four: Add existing files to your project	Summary
Add>>> Add>>> Add>>> Add>>> Add>>> Add>>> Add>>> Add>>> Remove Programes	Click 'Finish' to create/configure the project with these parameters. Project Parameters Device: PIC16F877A Toolsuite: Microchip MPASM Toolsuite File: D:\prog pic\TP1_mplab\TP1.mcp
récup D3 10G site simul base swnchropare Précédent Suivant> Annuler Aide	A new workspace will be created, and the new project added to that workspace.
< Précédent Suivant > Annuler Aide	< Précédent Terminer Annuler Aide

Ouvrir ensuite le répertoire que vous avez chargé (si la fenêtre de « navigation dans le projet » ne s'ouvre pas, faire « **View -> Project** ») et le sauvegarder, dans votre répertoire, sous un nom différent (Prog1_mplab.asm par exemple).

TP1 - MPLAB IDE v8.00 - Untitled Wor				
Project				
Output	Enregistrer sous			2 🛛 🗔
CPU Registers	Enregistrer dans :	UPI_mplab	⊻ 0 ⊅ ⊡•	
				at
TP1 - MPLAB IDE v8.00 - TP1.m				
File Edit View Project Debugger Pro				**
				**
TP1.mcw				i ç
E TP1.mcp				
PROG_1.ASM				**
	Nom du fichier :	PROG_1_mplab.ASM		Enregistrer
Clinker Script	Туре:	Assembly Source Files (*.asm,*.as,*.inc,*.s)	~	Annuler
	Jump to:	Project Directory	~	
	Encoding: ANSI	~		
Files Ptr Symbols	Ac	ld File To Project		.:

Par un clic droit dans le navigateur de projet sur le répertoire « **Sources Files** », accéder au menu contextuel, pour ajouter au projet le nouveau fichier que vous venez de sauvegarder. Supprimer l'ancien du projet (clic droit sur le nom).

🕅 TP1 - MPLAB IDE v8.00	TP1 - MPLAB IDE v8.00 - TP1.mcw	
File Edit View Project Debugger Programmer Tools Configure	File Edit View Project Debugger Programmer T	
🗋 🗅 😅 🖬 🐇 🐂 📾 🚔 🎒 🏙 🦉 📔 Release] D 😅 🖬 X 🐂 🛍 🛎 A 👐 💡	TP1.mcp*
Implance Implance Impla	TP1.mcw MPLAB I PROG_LAS Source Files PROG_LAS Assemble PROG_LAS Assemble Build Options Edit Remove Linker Script Other Files	Gource Files PROG_1_mplab.ASM Header Files Gibert Files Gibert Files Gibert Files Gibert Files Gibert Files Gibert Files

Par un double clic sur le nom du fichier restant, ouvrir celui-ci et analyser le code source.

2 <u>Compiler</u>

Compiler le fichier, la compilation devant se dérouler sans problèmes, comme l'indique la fenêtre « **Output** » de la figure suivante.



Si une erreur se produit lors de la compilation, un double clic sur la description de cette erreur dans la fenêtre « **Output** », ouvre le code source, le curseur se trouvant à l'endroit de l'erreur.

3 <u>Vérifier la mémoire programme</u>

MPLAB permet de visualiser se qui va être écrit dans la mémoire programme.

TP1 - MPI AB IDF v8.00	8.0	0 - Program N	iemory					
File Edit View Project Debugger Program	De	bugger Program	nmer Tools Confi	gure Wind	low Help			
Project Project	. 6			••• •• ••*	~ O 🛛		# 6	Checksum: 0xf21
	3 43		8 Reicas	5C 💌 🛄	2007 (and %)			
				🔜 Progr	am Memor	У		
□ IP1.1 Toolbars	-LA				Line	Address	Oncode	
CPULRegisters	3_1_	mplab.ASM			1	0000	2810	GOTO 0x10
Call Stack		goto	debut	7	2	0001	SEFE	ADDLW Oxff
Disassembly Listing					3	0002	3FFF	ADDLW Oxff
FEPROM		org Oxi	10		4	0003	3FFF	ADDLW Oxff
File Registers		debut			5	0004	3FFF	ADDLW Oxff
Flash Data		- TN	TTALTCATION		6	0005	3 F F F	ADDLW Oxff
Hardware Stack ni		.*********	**************		7	0006	3 F F F	ADDLW Oxff
LCD Pixel Di		-			8	0007	3 F F F	ADDLW Oxff
Locals		clrf	PORTB		9	0008	3FFF	ADDLW Oxff
Memory		bsf	STATUS, RPO		10	0009	3FFF	ADDLW Oxff
Program Memory A		bcf	STATUS, RP1		11	AOOO	3FFF	ADDLW Oxff
SER / Peripherals		movlw	Ъ'00000000'		12	OOOB	3FFF	ADDLW Oxff
Special Function Registers	-	movwrf	TRISB		13	0000	3FFF OFFF	ADDLW UXII
- Watch		DCI	SIRIUS, RPU		14	0000	STIT	ADDLW UXII
F		; PR(GRAMME PRINCI		16	DOOR	SEFE	ADDLW OXII
1 Memory Usage Gauge		;*******	******		17	0010	0186	CLRF OX6
<		boucle			18	0011	1683	BSF 0x3, 0x5
		bsf	PORTB,0		19	0012	1303	BCF Ox3, Ox6
		nop			20	0013	3000	MOVLW O
		nop	DODTR O		21	0014	0086	MOVWF 0x6
		goto	houcle		22	0015	1283	BCF 0x3, 0x5
		3000	204CLG		23	0016	1406	BSF Ox6, O
	1111				24	0017	0000	NOP
				Opcode H	Hex Machin	e Symbolic		
				1		1,0000		

Vérifier que les instructions du code sources sont bien aux adresses prévues.

Remarque : sur un emplacement mémoire non utilisé, tous les bits sont à 1 ce qui correspond à l'instruction de mnémonique « ADDLW 0xff ».

4 Simuler

Choisir ensuite le simulateur de MPLAB comme outil de simulation, et ouvrir une fenêtre de visualisation.

TP1 - MPLAB IDE v8.00	TP1 - MPLAB IDE v8.00	
File Edit View Project Debugger Programmer Tools Configure Window Help	File Edit View Project Debugger Program	
Select Tool None Clear Memory I Proteus VSM 2 MPLAB ICD 2 3 MPLAB ICD 2 3 MPLAB ICE 2000 Animate Hait F5 Step Into F7 Step Over F8 Step Out Reset Breakpoints F2 ON ON	Project Output Toolbars OUtput Toolbars OfU Registers Call Stack Disassembly Listing EEPROM File Registers Flash Data Hardware Stack LCD Pixel Locals Memory SFR / Peripherals Special Function Registers Watch I Memory Usage Gauge Simulator Trace Simulator Logic Analyzer Nop	Watch Add SFR CLATH Add SFR OPTION_REG OPTION_REG grabol Maxme OPCLATH PCLATH PON ITB OPEL2 ISB ON PIR1 ISB OPIR2 OPORTO PORTO PORTE PORTO PORTE PR2 Watch1 RCREG

Placer dans cette fenêtre tous les éléments intervenant dans notre programme comme indiqué ciaprès :

Watch												
Add SFR PCLATH	Add Symbol 16F877A	*										_
Address	Symbol Name	Value	Hex	Binary				_	_	_		
006	PORTB	0x00	0x00	00000000			ND	ւի	നി	ւրի	∃ ₽	
	WREG	0x00	0x00	00000000	~		~~	τs	U.	U.	LEU	
003	STATUS	0x18	0x18	00011000								
086	TRISB	0x00	0x00	00000000	1	2	2	Λ	5	6	7	
002	PCL	0x00	0x00	00000000		~	J	-	J	U	-	
00A	PCLATH	0x00	0x00	00000000								
Watch 1 Watch 2	Matala 2 Matala A											
watch / Watch 2	watch 5 watch 4											

Ajouter, par le menu contextuel de la barre du nom des colonnes (clic droit) une colonne « Hex » afin d'obtenir les valeurs en hexadécimal du contenu des registres.

Une barre d'outil permet de commander le simulateur :

- icône 1 : mode « Run », la simulation est exécuté à grande vitesse, jusqu'à un point d'arrêt ou l'appui sur l'icône « Halt » ;
- icône 2 : « Halt », arrête le mode « Run » et « Animate » ;
- icône 3 : mode « Animate », exécute le programme lentement jusqu'à l'appui sur « Halt » ;
- icône 4 : mode « **Step Into** », exécute le programme pas à pas en passant dans tous les sous programmes ;
- icône 5 : mode « Step Over », exécute le programme pas à pas sans passer dans les sous programmes ;
- icône 6 : mode « Step Out », exécute le programme pas à pas en sortant des sous programmes ;
- icône 7 : mode « Reset », place le compteur programme à l'adresse 0x0000.

Remarques :

- si on souhaite initialiser la valeur d'un registre, il suffit de cliquer deux fois sur celle-ci, et entrer la nouvelle valeur ;
- il est possible, pour des programmes plus complexes, de définir jusqu'à 4 fenêtres de visualisation.

Lancer une simulation en mode pas à pas et observer l'évolution des différents registres.

MPLA	D IDE Editor	L						
PROG_1_	_mplab.ASM		×					
	bsf	STATUS, RPO		Watch				
	bcf movlw	STATUS, RP1 b'00000000'	- * : **	Add SFR ADCON0	Add Symbol 666877A	~		
	movwf	TRISB	- 2	Address	Symbol Name	Value	Hex	Binary
	bcf	STATUS, RPO	;	006	PORTB	0x00	0x00	00000000
	, . DD(COMME DOTHETD	δ.T.		WREG	0x00	0x00	00000000
	********	**********	****	003	STATUS	0x1C	Ox1C	00011100
	houcle			086	TRISB	0x00	0x00	00000000
	bsf	PORTB.0		002	PCL	0x16	0x16	00010110
	nop	,	2 =	001	PCLATH	0x00	0x00	00000000
	bcf	PORTB, O	2.1					
	goto	boucle	; -					
	END		;	Watch 1 Watch 2	Watch 3 Watch 4			

On notera que la ligne pointée par la flèche verte dans le programme n'est pas la ligne en cours d'exécution, mais celle qui va être exécuté au cycle suivant, de même que PCL pointe l'adresse qui va être exécutée et non celle en cours d'exécution.

4.1 Une erreur classique...

On a pu constater que le programme précédent permettait de faire basculer la sortie PB0. De nombreuses erreurs de programmation avec les microcontrôleurs PIC sont dues à une écriture dans la mauvaise banque : nous aurions pu oublier par exemple de revenir en banque 0 dans notre programme après la phase d'initialisation du port B.

Observons ce qui ce passe alors en plaçant un « ; » devant la ligne suivante du programme source :

bcf	STATUS, RP0	; retour en banque 0	
compiler, sim	nuler et observer les d	fférences.	

5 <u>Le mode débogage</u>

Le mode débogage permet de simuler le pic en situation, tout en gardant le contrôle du microcontrôleur (mode pas à pas, observation de registres etc...).

Cette solution est moins onéreuse que l'émulateur, mais utilise certaines ressources du circuit cible :

- des ressources matérielles bornes PGC et PGD (qui sont aussi RB6 et RB7 sur le PIC 16F877A) ainsi que Vpp/MCLR ;
- des ressources logicielles : implantation d'un programme en mémoire haute, utilisation de registre, de la pile.

Le chien de garde, la protection contre la relecture, la protection de relecture de l'EEPROM doivent être de plus désactivés, ainsi que la programmation basse tension; l'oscillateur doit avoir la configuration correspondant à la cible.

Si ce n'est pas encore fait :

- connecter l'ICD 2 au port USB du PC ;
- connecter la carte "PICDEM 2 PLUS à l'ICD2 ;
- alimenter la carte par le bloc secteur.

Il est possible que le système d'exploitation souhaite installer les pilotes de l'ICD2.

Le fichier demandé « icd2w2kl.inf » se trouve dans le répertoire C:\Program Files \ `Microchip \ MPLAB IDE \ ICD2 \ Drivers ».

Sélectionner dans un premier temps l'ICD2 comme débogueur, la fenêtre « Output » devant signaler qu'elle a bien reconnu ce module.



En cas de problème, on pourra vérifier que :

- le circuit installé sur la carte est bien celui déclaré au logiciel ;
- la connectique est correcte, la carte correctement alimentée ;
- les paramètres sont corrects, en particuliers en ce qui concerne l'alimentation de la carte (qui n'est pas alimentée depuis l'ICID2 dans notre cas).

8.00		<u>1</u>				-0.101	
Debugger Programmer Tools Configure W	MPLAB ICD Z Settings				0 2 Settings		
Select Tool	Versions	Warnings	Free	V	ersions	Warnings	Freezo
Clear Memory	Status Communica	ition Limitations	Power	Status	Communication	Limitations	Power F
Run F9							
Animate	Target Vdd	5.08		Conr	ect Status		
Halt F5	Target Vpp	12.54 Update		Con	nected		
Step Over E8	MPLABICD 2 Von	12.54	5.4		utomatically connect a	it startup	
Step Out		in a state of the		1 🔽 A	utomatically download	firmware if needed	
Reset +	Values are updated on o	onnection, requests to		Mes	ages		
Breakpoints E2	update, and power settir	ng changes			utput to debua file		
Advanced Breakpoints	Power target circuit f	rom MPLAB ICD 2 (5V Vdd)	2				
MDLAB ICD 2 Setup Witzerd			8	Self	lest		
				i arg			
Program				• Mod	le Vpp Pass		
Read Read FERROM				MCL	R Gnd Pass	Run Self Test	
				4 MCL	R Vdd Pass		
Abort Operation				MCL	R Vpp Pass		
Connect							
Download ICD2 Operating System							
Settings							

Il faut ensuite préciser que la compilation à effectuer est destinée au mode « débogage » (un morceau de programme va être ajouté en partie haute de la mémoire programme, afin de contrôler le circuit cible).

Compiler ensuite, puis programmer (attention depuis le menu « Debugger » et non « Programmer ») le circuit cible.

PLA	B IDE v8.00			TP1 - MPLA	3 IDE v8.00			8.00	
зw	Project Debugger Programmer	r Tools Configure Window	File	le Edit View	Project Debugger Progra	ammer Tools	Confiç	Debugger Programmer To	ols Configure W
1	Project Wizard				Project Wizard		5	Select Tool	• •
	New				New			Clear Memory	· · · ·
w	Open			TP1.mcw	Open			Run	F9
	Close	•		-	Close		• •	Animate	
Lm	Set Active Project			🖃 🛄 TP1.m	Set Active Project		→ L	Halt	F5
S.	Quickbuild (no., asm file)			🖻 🛄 Sou	Quickbuild (noasm file)		Ĩ	Step Into	F7 🞽
	Quictodila (no rasimilio)	X		<u></u>	Quero dalla (no resin moy		—I	1 Step Over	F8 🗙
н	Clean			🧰 Hea	Clean		,	- Step Out	
0	Export Makefile			🚞 Obj	Export Makefile			Reset	F 🔼
Li	Build All	Ctrl+F10 Add		🚞 Libr	Build All	Ctrl+F10)	Breakpoints	E2
Li	Make	F10 ; F10		🚞 Link	Make	F10	2	Advanced Breakpoints	
0	Build Configuration	Release	-	- 🗀 Oth	Build Configuration				
	Build Options	🕨 🗸 Debug			Build Options		<u> </u>	MPLAB ICD 2 Setup Wizard	
utp	Save Project				Save Project		7	Program	
	Save Project As				Save Proiect As		_ L	C Read	
	Add Files to Proiect		-					Read EEPROM	
								Abort Operation	

La démarche est ensuite identique à la simulation, mais c'est le circuit cible qui fonctionne réellement.

Vérifier en mode pas à pas que l'on a bien le fonctionnement attendu et que la DEL s'allume et s'éteint.

6 Programmer

Pour programmer le circuit cible, il faut d'abord préciser au logiciel quel programmateur on va utiliser (ici l'ICD2), vérifier que le programmateur est bien détecté (fenêtre « **Output** »).

En cas de problème, on pourra vérifier que :

- le circuit installé sur la carte est bien celui déclaré au logiciel ;
- la connectique est correcte, la carte correctement alimentée ;
- les paramètres sont correctes dans « **Programmer -> Settings** », en particuliers en ce qui concerne l'alimentation de la carte (qui n'est pas alimentée depuis l'ICID2 dans notre cas) ;
- le pilote « icd2w2kl.inf » de l'ICD2 est installé (voir chapitre précédent).

		Output	B IDE v8.00		
Programmer Tools Configure Windo	w Help		Project Debugger Programmer	Tools Con	figure Windo
Programmer Tools Configure Windd Select Programmer > MPLAB ICD 2 Setup Wizard > Program Read > Verify Erase Part > Blank Check Read EEPROM >	W Help None 1 PICSTART Plus 2 MPLAB ICD 2 3 ANBS1 Quick Programmer Beta 4 PICkit 2 5 MPLAB PM 3 6 REAL ICE 7 PRO MATE II 8 PICkit 1	Output Build Version Control Find in Files Connecting to MPLAB ICD 2 Connected Setting Vdd source to target Target Device PIC16F877A founc Reading ICD Product ID Running ICD Self Test Passed MPLAB ICD 2 Ready	Project Debugger Programmer Project Wizard New Open Close Set Active Project Quickbuild (no .asm file) Clean Export Makefile Build Configuration Build Options Save Project	Tools Con , , , , , , , , , , , , ,	figure Windo se v if (figure windo) se v if (figure windo) figure windo) figure windo) figure windo) figure windo) se v if (figure windo) se v if (
			Save Project As Add Files to Project		ont nécess

On précisera également que la compilation doit se faire maintenant en mode « $\ensuremath{\textit{Release}}$ » et non « Debug »

Déconnecter le cordon RJ45 de la carte PICDEM 2 PLUS ; la DEL clignotant à fréquence élevée, elle semble allumée en permanence et il n'est pas possible de constater à l'œil le clignotement. Si nous souhaitons voir clignoter la DEL, il va falloir diminuer la fréquence de commande.

7 Modification du programme

Afin de ralentir la fréquence de la sortie RB0 nous allons inclure dans notre programme une boucle de temporisation.

La version modifiée est disponible dans le dossier « **Ressources / TP1 mplab / Prog2.asm** ». On trouvera ce programme reproduit en annexe 2.

Ajouter ce programme au projet et supprimer l'ancien du projet (il reste malgré tout disponible dans notre répertoire).

Etudier ce nouveau programme.

On notera en particuliers :

- l'utilisation de la directive « #DEFINE » pour remplacer « PORTB,0 » par « DEL »
- la déclaration de variables en RAM par la directive « EQU » qui remplace T_INT et T_EXT dans le programme, respectivement par 0x20 et 0x21 comme nous l'avons vu pour le fichier « include » ; attention 0x20 et 0x21 sont les adresses de T_INT et T_EXT et non leur contenu.
- l'appel de sous programme pour l'initialisation du port et la temporisation, ce qui améliore la lisibilité de l'ensemble ;

Compiler le programme en mode « réalisation » par « Project -> Build Configuration -> Release » puis « Ctrl F10 ».

Programmer le composant et vérifier que le fonctionnement du microcontrôleur est bien celui attendu.

Revenons maintenant en mode simulateur, afin d'étudier quelques fonctionnalités avancées de celuici.

7.1 Les SFR et la pile

Préparer la simulation en ajoutant T_INT et T_EXT dans la fenêtre « **Watch** » comme nous l'avons vu précédemment.

Ouvrir ensuite des fenêtres représentant le plan mémoire des fichiers et données, et enfin le contenu de la pile.

Watch		🐼 TP1 -	MPLAB IDE v8.00 - Watch	- 21	MPLAB IDE v8.00 - Watch
Add SFR ADCON0 🗸 Add Symbol 🚺		File Edit	View Project Debugger Progr	aı Edit	View Project Debugger Progr
Address Symbol NS	SPM3 SPOV	🗋 🗅 🚅	 ✓ Project ✓ Output 	1 🖻	 ✓ Project ✓ Output
006 PORTB S WREG T	YNC _EXT	T P1.	Toolbars +	P1.	Toolbars 🕨
003 STATUS T	INT OCS		CPU Registers		CPU Registers
000 PCL	OIE OIF		Call Stack	-	Call Stack
OOA PCLATH T	0SE 1CKPS0		EEPROM		EEPROM
			File Registers		File Registers
Ť			Hash Data Hardware Stack	*	Hash Data Hardware Stack
Watch 1 Watch 2 Watch 3 Watch 4	2CKPS0				LCD Pixel
					Locals .
					Program Memory

Réinitialiser le programme par un « **Reset** » puis jouer sur les options « **Step Into** », « **Step Over** » et « **Step Out** ».

				Add SEB	ADCONO	V [A	dd Sum	hol	16F87	74						~
	; temps total du sous	programm					idd oyin				- 1			Dána	200. 2010	× 1
	tempo			Add	ress	2	ogmoo	i Nai	ne	v	arue	ne	×	bina	гy	
	movlw h'FF'			C	10.6	PO	RTB				0x01	OXI	31	0000	0001	
	movwf T EXT	, ,		3		WR	EG				OxFF	Ox1	FF	1111	1111	
	boucle ext	· · ·		6	103	ST	ATUS				0x10	OX:	1C	0001	1100	
	movlw h'FF'				186	TR	ISB				0x00	Oxt	00	0000	0000	
	movwf T INT			C	102	PC	L				0x24	Ox:	24	0010	0100	
	boucle_int	· · · · · · · · · · · · · · · · · · ·		C	IOA	PC	LATH				0x00	Oxt	00	0000	0000	
	decfsz T_INT,1	2		0	121	т_	EXT				OxFF	OxI	FF	1111	1111	
	goto boucle_int	2 2		C	20	т_	INT				OxFI	OxI	FD	1111	1101	
1.00	decfsz T_EXT,1															
	goto boucle_ext	2 A 🚺		Match 1	have all		. The second									
	return			waten i	Watch 2	Watch .	3 Wat	ch 4								_
															1010000	
					File Regist									1		\sim
lardw	are Stack - ICD2: View no	t supported		3 🚽	File Regist	ers no c	1 02	03	04	15 06	07 0	8 09	0.1	OB OC		
lardw TOS	are Stack - ICD2: View no Stack Level Re	t supported	Location		ddress	ers 00 0	0 24	03	04	05 06	07 0	8 09	01	0B 00		^
lardw TOS	are Stack - ICD2: View no Stack Level Re	<mark>t supported</mark> turn Address Empty	Location		File Regist ddress 000 010	ers 00 0 0	1 02 0 24	03 1C	04	05 06 00 01	07 C	8 09 0 00	0 A 0 0 0	0B 00		^
Hardw TOS	are Stack - ICD2: View no Stack Level Re 0	t supported turn Address Empty 0013	Location		ddress 000 010 020	ers 00 0 0 00 0 FD F	1 02 0 24 0 00 F 00	03 1C 00	04 00 00	05 06 00 01 00 00	07 0 00 0 00 0	8 09 0 00 0 00	00 00 00	ОВ 00 00 00 00 00		
lardw TOS ₽	are Stack - JCD2: View no Stack Level Re 0 1 2	t supported turn Address Empty 0013 0000	Location		ddress 000 010 020 030	ers 00 0 0 00 0 FD F	1 02 0 24 0 00 F 00	03 1C 00 00	04 00 00 00	D5 O6 D0 01 D0 00 D0 00 D0 00	07 0 00 0 00 0 00 0	8 09 0 00 0 00 0 00	0A 00 00 00	0B 00 00 00 00 00 00 00	C OD 0 00 0 00 0 00 0 00	
Hardw TOS	are Stack - ICD2: View no Stack Level Re 0 1 2 3	t supported turn Address Empty 0013 0000	Location		File Regist ddress 000 010 020 030 040	ers 00 C 0 00 0 FD F 00 0	1 02 0 24 0 00 F 00 0 00	03 1C 00 00 00	04 00 00 00 00	D5 O6 00 01 00 00 00 00 00 00 00 00 00 00	07 0 00 0 00 0 00 0 00 0	8 09 0 00 0 00 0 00 0 00	00 00 00 00	0B 00 00 00 00 00 00 00 00 00	C OD 0 00 0 00 0 00 0 00 0 00	
lardw TOS ₽	are Stack - ICD2: View no Stack Level Re 0 1 2 3 4	t supported turn Address Empty 0013 0000 0000	Location		File Regist ddress 000 010 020 030 040 050	ers 00 0 00 0 FD F 00 0 00 0	1 02 0 24 0 00 F 00 0 00 0 00	03 1C 00 00 00 00	04 00 00 00 00 00	D5 O6 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00	07 0 00 0 00 0 00 0 00 0 00 0	8 09 0 00 0 00 0 00 0 00 0 00	00 00 00 00 00 00	0B 00 00 00 00 00 00 00 00 00 00 00	C OD 0 00 0 00 0 00 0 00 0 00 0 00	
Hardw TOS	are Stack - ICD2: View no Stack Level Re 0 1 2 3 4 5	t supported turn Address Empty 0013 0000 0000 0000 0000	Location		File Regist ddress 000 010 020 030 040 050	ers 00 0 0 00 0 FD F 00 0 00 0 00 0 00 0	1 02 0 24 0 00 F 00 0 00 0 00 0 00	03 1C 00 00 00 00 00	04 00 00 00 00 00 00	D5 O6 00 01 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	07 0 00 0 00 0 00 0 00 0 00 0	8 09 0 00 0 00 0 00 0 00 0 00 0 00	0Å 00 00 00 00 00 00	0B 00 00 00 00 00 00 00 00 00 00 00 00 00	D D D D D D D D	

Observer dans un premier temps l'évolution de T_INT et T_EXT, à la fois dans la fenêtre « **Watch** » et dans le plan mémoire des registres « **File Registers** », ainsi que le contenu du compteur de programme PC (adresse 02 du plan mémoire) en vérifiant qu'il est identique à PCL pour ce petit programme.

Observer ensuite l'évolution du contenu de la pile et de PC lors des appels de sous programmes.

7.2 Les points d'arrêts

Lorsque le simulateur fonctionne en mode « **Run** », il est possible de placer des points d'arrêts dans le programme en double cliquant sur n'importe quelle ligne exécutable. La simulation s'arrêtera alors sur ce point lorsqu'elle y passera (si elle y passe !). Cet outil est très pratique pour vérifier qu'un programme passe bien par une sous-routine.

boucle_ext movlw h'FF' movvf Tl boucle_int	; l cycle ; l cycle, initialisation de Tl à FF
decfsz T1,1	; l cycle decrementation de Tl : 2 cycles
Add Filter-in Trace Add Filter-out Trace Remove Filter Trace Remove All Filter Traces	; 1 cycle decremente T2 ; 2 cycles
Close Remove Breakpoint Disable Breakpoint	; directive signalant la fin du programme
Breakpoints	

Par le menu contextuel (clic droit), il est ensuite possible d'enlever complètement le point d'arrêt, qui disparaît alors (un double clic a le même effet), ou de le désactiver, il reste alors un cercle rouge.

Tester ce mode de fonctionnement en plaçant un point d'arrêt dans la temporisation et lancer le mode « Run ».

Supprimer les points d'arrêts avant de passer au paragraphe suivant.

7.3 <u>Le mode Trace</u>

Ce mode permet d'enregistrer les actions du programme et le contenu des registres.

8.00	Simulator Settings		MPLAB IDE v8.00
Debugger Programmer Select Tool Clear Memory	Code Coverage Animation / Realtime Updates Limitat Osc / Trace Break Options SCL Options Uart1		View Project Debugger F Project Output
Run Animate Halt Step Into Step Over Step Out Reset	Processor Frequency Units: MHz KHz Hz Trace Options	Run	Toolbars ⊂ CPU Registers ⊂ Call Stack Disassembly Listing EEPROM File Registers Elach Data
Breakpoints StopWatch Complex Breakpoints Stimulus Profile Clear Code Coverage Refresh PM	Trace All Buffer Size (1K - 45590K) Break on Trace Buffer Full Buffer Full OK Annuler Apple	Output Build Version Control Find in Files MPLAB SIM MPLAB CORE-W0011: Program halted. Trace Buffer Full CORE-W0011: Program halted. Trace Buffer Full	Hardware Stack LCD Pixel Locals Memory Program Memory SFR / Peripherals Special Function Registers Watch
Settings			1 Memory Usage Gauge Simulator Trace Simulator Logic Analyzer

Dans le fenêtre de configuration de l'oscillateur, cocher l'option « Trace All » et choisir l'enregistrement jusqu'à ce que le tampon soit plein, à 1 K lignes (ce qui est suffisant pour notre programme).

Lancer une simulation, qui va être arrêté au bout de 1024 lignes d'exécution. Ouvrir la fenêtre « **Simulator Trace** ».

premiers pas avec MPLAB 8.0

DOC 2	molab ACM			Trace									
nou_2_			<u> </u>	Line	Addr	Op La	abel Instruction	1 SA	SD	DA	DD	Cycles	2
	; TEM	PURISATION		0	0000	2810	GOTO 0x10				-	100000000	
	. Remark r=4	MHr. Torrales	4 /Remove	1	0001	0000	NOP		(-,-))00000001	
	; boucle in	terne :768 u	s again	2	0010	2018 deb	out CALL Ox18	()				300000005	
	; boucle ex	terne : (bou	cle int	3	0011	0000 bou	ucle NOP	()				100000003	
	; temps tot	al du sous pi	rogramm	4	0018	0186 ini	it CLRF Ox6	()		0006	00	300000004	
	14-19-19-19-19-19-19-19-19-19-19-19-19-19-		A DESCRIPTION OF THE PARTY OF T	5	0019	1683	BSF 0x3, 0x5	0003	10	0003	3.0	100000005	
	tempo			6	001A	1303	BCF 0x3, 0x6	0083	3C	0003	3C	300000006	
	movlw	h'FF'		7	001B	3000	MOALM 0	W		W	00	100000007	
	movwf	T_EXT	2	8	001C	0086	MOVWF Ox6		57	0086	00	30000008	
	boucle_ext			9	001D	1283	BCF 0x3, 0x5	0083	3 C	0003	1C	300000009	
	movlw	h'FF'	2	10	001E	0008	RETURN		-			10000000A	
	movwf	T_INT	÷ .	11	001F	0000 ten	npo NOP)0000000B	
	boucle_int			12	0011	1406 bou	acle BSF Ox6, O	0006	00	0006	01	3000000C	
	decfsz	T_INT,1	7	13	0012	201F	CALL Ox1f					1000000D	
	goto	boucle_int	2 m	14	0013	0000	NOP		-)0000000E	
	dectsz	T_EXT,1		15	001F	30FF ten	npo MOVLW Oxff	W		IJ	FF)0000000F	
	goto	boucle_ext	7 35	16	0020	00A1	MOVWF 0x21			0021	FF)00000010	
	recurn			17	0021	30FF bou	acle MOVLW Oxff	W	. 	IJ	FF)00000011	
				18	0022	OOAO	MOVWF 0x20			0020	FF)00000012	
	END			19	0023	OBAO bou	acle DECFSZ Ox20,	F0020	FF	0020	FE	300000013	
	Strength .		2	20	0024	2823	GOTO 0x23					300000014	
				21	0025	0000	NOP)00000015	
				22	0023	OBAO bou	ucle DECFSZ Ox20,	F0020	FE	0020	FD	300000016	
			12201	23	0024	2823	GOTO 0x23)00000017	
1400.00			×		0005		MOR					200000010	
<u><</u>			>	D:\PROG	PIC\TF	P1_MPLAB	\PROG_2_MPLAB.ASI	M					
10				59		clrf 1	PORTB ;	RAZ de:	s ba	scule	зD		1
outpu	II.			60		bsf :	STATUS, RPO			1922			
Build V	/ersion Control F	ind in Files MPL4	AB SIM MPLAB	61		bef ;	STATUS, RP1 ;	passage	e en	banq	ue O		
CODEN	W0011, Due euro	n halfad. Tura	- Duffen Full	62		movlw 1	b'00000000'	5) (7)		15			-
CORE-V	W0011. Progral	minalled. Trace	e Duiler Full	63		movwf !	TRISB ;	PORTB e	en s	ortie			
UURE-V	wuuii: Progra	minaited. Trace	e Buπer Full	2	a	3002030T		100.001070					

Un double clic sur une ligne de la fenêtre « Trace » ouvre une sous fenêtre avec la ligne de programme correspondant.

Dans la fenêtre « Trace » on peut lire :

- le numéro de la ligne de programme concernée ;
- l'adresse « Addr » de la mémoire programme à laquelle nous sommes ;
- le code « **Op** » de l'opération ;
- l'étiquette s'il y en a une ;
- l'instruction
- l'adresse du registre sources « SA » ;
- la donnée « SD » contenue dans ce registre avant l'opération ;
- l'adresse « **DA** » du registre destination ;
- la donnée « **DD** » contenue dans le registre destination après l'opération.

7.4 Mesure de temps

Il est également de mesurer le temps entre deux instructions avec la fenêtre « Stopwatch ».



- vérifier dans un premier temps que le simulateur est bien paramètré pour un quartz de 4 MHz (« Debugger -> Settings -> Osc / Trace »);
- ouvrir la fenêtre « Stopwatch » par le menu « Debugger » ;
- réinitialiser la simulation, ce qui réinitialise les valeurs de la fenêtre ;
- placer le curseur dans le programme, à l'endroit où l'on souhaite démarrer la mesure de temps, puis par le menu contextuel (clic droit), faire « Run to the cursor » ; la fenêtre « Stopwatch » affiche alors le nombre de cycle d'instructions écoulées depuis le reset et le temps passé (dans notre cas 1 µs par cycle) ;
- initialiser la fenêtre « Stopwatch » par « Zero » ;
- placer le curseur dans le programme à l'endroit où l'on souhaite terminer la mesure, puis amener la simulation jusqu'à cette ligne ;
- la fenêtre « Stopwatch » affiche alors dans la première colonne la durée depuis le « Zero » (c'est à dire la durée de notre temporisation), ainsi que la durée depuis le début du programme dans la colonne de droite ;
- en cliquant sur « Synch » il est possible de resynchroniser la première colonne sur la seconde.

7.5 Analyseur logique

Il est possible de visualiser les signaux sous forme de chronogrammes.

Pour éviter une simulation trop longue, modifier le programme pour que la fréquence de RB0 soit environ 1,5 ms puis compiler de nouveau.

Vérifier que la configuration du simulateur correspond bien à une fréquence de quartz de 4 MHz et imposer une taille de tampon de 5 K lignes.

Ouvrir l'analyseur logique et ajouter le signal RB0 dans la sélection.

Simulator Setting	s	alt e	? 🛛	- MPLAB IDE v8.00	rogra				
Simulator Setting Code Coverage Osc / Trace Processor Freque Trace Options Trace All Break on Tr	Animation / Re Break Options ency ace Buffer Full OK	ealtime Updates SCL Options International SCL Options International Inte	Limitations Uart 10 Hz Hz z - 45590K)) K lines) M lines	MICAB IDE Vo.00 View Project Voltut Toolbars CPU Registers Call Stack Disassembly Listing EEPROM File Registers Filash Data Hardware Stack LCD Pixel Locals Memory Program Memory SFR / Peripherals Special Function Registers Watch	rogra state 1 1 1 1 1 1 1 1 1 1 1 1 1	Available Signals All VCP2 INTO RA1 RA2 RA3 RA4 RA5 RB1 RB2 RB3	els Configure Bus(s) Add => Remove <=	Selected Signal(s) RB0	Move Up Move Down
				1 Memory Usage Gauge Simulator Trace Simulator Logic Analyzer		RB4 RB5 RB6	Ca	ncel	Help

Lancer la simulation en mode « Run » ; elle s'arrête dès que le tampon est plein.

A l'aide des curseurs, vérifier que la période est bien celle attendue, l'affichage se faisant en « temps de cycle », qui vaut un quart de la période du quartz d'horloge, soit 1 µs.

premiers pas avec MPLAB 8.0



7.6 Simulation d'évènements externes

On souhaite maintenant que la DEL ne clignote que si on appui sur le bouton poussoir S2 de la carte, ce qui aura pour effet de mettre à 0 le bit 4 du port A.

Modifier le programme (on trouvera un exemple de solution en annexe 4) :

- en déclarant une variable BP liée au bit 4 du port A par « #DEFINE » ;
- en configurant en entrée ce bit (consulter la documentation) ;
- en réalisant le test de ce bit par une instruction du type « btfsc » ou « btfss ».

Introduire un « **stimuli** » et sur l'onglet « **Asynch** » de la fenêtre qui s'ouvre (notre événement sera à priori asynchrone) :

- par un clic gauche dans la colonne « **Pin/SFR** » on accède par un menu déroulant aux différentes entrées ;
- par un clic gauche dans la colonne « Action », on choisit l'action liée à cette entrée ;
- il suffit ensuite, au moment de la simulation, de cliquer sur la colonne « Fire » pour déclencher l'action correspondante.



Simuler votre et programme et le tester sur le microcontrôleur une fois qu'il est au point.

Annexe 1 : Premier programme

; Ce programme génère une horloge en RB0 ; de fréquence 24 fois plus faible que celle du quartz. * ; il utilise la carte PicDem2Plus équipée d'un PIC16F877A * LIST P=16F877A ; directive qui définit le processeur utilisé									
LIST P=16F877A ; directive qui définit le processeur utilisé									
#include <p16f877a.inc> ; fichier de définition des constantes</p16f877a.inc>									
; BITS DE CONFIGURATION									
, CONFIGHS_OSC &WDT_OFF &CP_OFF &CPD_OFF &LVP_OFF									
; _XS_OSC l'oscillateur est configuré en oscillateur à quartz haute fréquence ; _WDT_OFF le watchdog est désactivé ; _CP_OFF le code de protection de la mémoire programme est désactivé ; _CPD_OFF le code de protection de la mémoire EEPROM est désactivé ; _LVP_OFF la programmation basse tension est désactivée ; ces opérations sont nécessaires pour fonctionner en mode "debug"									
,									
org 0x0 ; Adresse de départ après reset goto debut									
org 0x10 ; adresse de début du programme debut									
; INITIALISATION *									
; initialisation du PORTB en sortie (voir datasheet)									
bcfSTATUS, RP0bcfSTATUS, RP1; passage en banque 0clrfPORTB; RAZ des bascules DbsfSTATUS, RP0; passage en banque 1movlwb'00000000'movwfTRISB; PORTB en sortiebcfSTATUS, RP0; retour en banque 0									
·*************************************									
; PROGRAMME PRINCIPAL *									
, boucle bsf PORTB,0 ; mise à 1 de la sortie nop ; 2 temps morts pour compenser le saut nop bcf PORTB,0 ; mise à 0 de la sortie goto boucle ; rebouclage									
END ; directive signalant la fin du programme									

Annexe 2 : Second programme



LIST P=16F877A ; directive qui définit le processeur utilisé #include <P16F877A.INC> **BITS DE CONFIGURATION** _CONFIG _HS_OSC & _WDT_OFF & _CP_OFF & _CPD_OFF & _LVP_OFF ; _XS_OSC l'oscillateur est configuré en oscillateur à quartz haute fréquence ; _WDT_OFF le watchdog est désactivé ; _CP_OFF le code de protection de la mémoire programme est désactivé CPD_OFF le code de protection de la mémoire EEPROM est désactivé ; LVP_OFF la programmation basse tension est désactivée ; ces opérations sont nécessaires pour fonctionner en mode "debug" DEFINITIIONS **#DEFINE DEL PORTB.0** ;la DEL sur le port B, bit 0 **** **DECLARATION de VARIABLES** ***** 0x20 T INT EQU ; Variable de temporisation à l'adresse 0x20 EQU 0x21 ; Variable de temporisation à l'adresse 0x21 T_EXT

****** DEMARRAGE SUR RESET ***** 0x0 org ; Adresse de départ après reset debut goto 0x10 ; adresse de début du programme org debut PROGRAMME PRINCIPAL ***** ***** call init ; l'initialisation du port se fait par sous programme boucle bsf DEL ; mise à 1 de la sortie call tempo DEL ; mise à 0 de la sortie bcf call tempo goto boucle ; rebouclage ***** SOUS PROGRAMMES ***** ********** INITIALISATION ***** ; initialisation du PORTB en sortie (voir datasheet) init bcf STATUS, RP0 STATUS, RP1 ; passage en banque 0 bcf clrf PORTB ; RAZ des bascules D STATUS, RP0 ; passage en banque 1 bsf movlw b'00000000' ; PORTB en sortie movwf TRISB STATUS, RP0 ; retour en banque 0 bcf return TEMPORISATION ; Fquartz=4MHz, Tcycle=4/Fquartz=1µs ; boucle interne :environ 3 µs .255= 765 µs ; temps total du sous programme : environ 255.765 µs, environ 200 ms tempo movlw h'FF' ; initialisation de T_EXT à FF movwf T_EXT boucle_ext movlw h'FF' ; 1 cycle movwf T_INT ; 1 cycle, initialisation de T_INT à FF boucle_int decfsz T_INT,1 ; 1 ou 2 cycles, décrémentation de T_INT goto boucle_int ; 2 cycles decfsz T_EXT,1 ; 1 ou 2 cycles, décrémentation T EXT goto boucle ext ; 2 cycles return ; directive signalant la fin du programme END

Annexe 3 : les ports A et B

Les ports d'entrées sorties sont configurables bit à bit en entrée ou en sortie par le registre TRIS du même nom que le port considéré : l'écriture d'un NL0 sur le bit x du registre TRISY configure le bit x du PORTY en sortie (« 0 » comme « Output », c'est la valeur par défaut après une réinitialisation) tandis qu'un NL1 le configure en entrée (« 1 » comme « Input »).

Particularité du port A :

- il s'agit d'un port 5 bits ;
- les bornes 0 à 3 sont multiplexées avec de fonctionnalités du CAN.
- la sortie 4 est à drain ouvert
- la borne 4 est multiplexée avec la fonction « entrée d'horloge du timer 0 » et sortie du « comparateur » ;
- la borne 5 est multiplexée avec une fonction entrée du CAN, entré pour la liaison série synchrone, et sortie du comparateur.

Particularité du port B :

- il s'agit d'un port 8 bits ;
- sur chaque borne il est possible de configurer par logiciel une résistance de tirage vers le haut ;
- le bornes 3, 6 et 7 sont multiplexée pour la programmation ;
- les entrées 4 à 7 peuvent servir d'entrée d'interruption ;

Pour configurer une borne en entrés sortie numérique, le constructeur préconise :

- de réinitialiser le port ;
- de configurer les bornes en entrées numérique s'il y a un multiplexage avec une fonction analogique ;
- d'écrire dans le registre TRIS ;

Ci-après un exemple pour le port A ;

BCF	STATUS,	RPO	1
BCF	STATUS,	RP1	; Bank0
CLRF	PORTA		; Initialize FORTA by
			; clearing output
			; data latches
BSF	STATUS,	RPO	; Select Bank 1
MOVLW	0x06		; Configure all pins
MOVWF	ADCON1		; as digital inputs
MOVLW	0xCF		; Value used to
			; initialize data
			; direction
MOVWF	TRISA		; Set RA<3:0> as inputs
			; RA<5:4> as outputs
			; TRISA<7:6>are always
			; read as '0'.

Annexe 4 :exemple de solution

Exemple de solution pour la partie « simulation d'événements externes » ; Seules les parties à changer par rapport au programme précédent ont été reproduites.

.*******	*******		***************************************
, .******	*******		****
, #DEFII		PORTB.0	:la DEL sur le port B. bit 0
#DEFI	NE BP P	ORTA,4	;la DEL sur le port B, bit 0
•******* ,	*******	*****	*****
;		PROGRAMME PRINC	IPAL *
,*******		*****	*************************
boucle	call Init		
Doucle	btfsc	BP	· test du bouton poussoir
	aoto	boucle	: bouclage sur le test si BP=1
	bsf	DEL	; mise à 1 de la sortie
	call	tempo	
	nop		; 2 temps morts pour compenser le goto
	nop		
	bcf	DEL	; mise à 0 de la sortie
	call	tempo	· robouologo
	golo	Doucle	, rebouciage
.******	******	*****	******
;		SOUS PROGRAMMES	*
·*******	******	*****	******************************
.******* ,	*******	*******************************	***************************************
• , • • • • • • • • • •	****	INITIALISATION	*
,	*****	*****	visitialization du DODTD an partia (voir datashaat)
			; Initialisation du PORTE en sonie (voir datasneet)
init			
mitt	bcf	STATUS, RP0	
	bcf	STATUS, RP1	; passage en banque 0
	clrf	PORTB	; RAZ des bascules D du port B
	clrf	PORTA	; RAZ des bascules D du port A
	bsf	STATUS, RP0	; passage en banque 1
	movlw	b'00000000'	
	movwf	TRISB	; PORTB en sortie
	moviw		
	moviw	ADCONT 6'00010000'	, PORTA conligure en numenque
	movwf	TRISA	· PORTA bit 4 en entrée
	bcf	STATUS, RP0	; retour en banque 0
	return		