

TP 3 : Mise en œuvre d'un capteur de température I2C DS1621.

Encadrants : <u>Hélène LEYMARIE</u> // <u>Thierry PERISSE</u>

Objectifs:

Afficher sur un LCD la valeur de la température donnée par un capteur I2C (DS1631)

Savoir faire : (pré-requis)

Savoir utiliser le logiciel "mikroC PRO for PIC" permettant de programmer un microcontrôleur PIC16F877 avec un langage de haut niveau (langage C).

Savoir utiliser le logiciel de simulation de schéma électrique "PROTEUS"

Savoir comprendre une communication I2C

Outils logiciels :





Procédure <u>Téléchargement PROTEUS</u> <u>demo</u>

Procédure <u>Téléchargement mikroC</u> <u>PRO for PIC</u>

Matériels hardwares utilisés

Aa Composants	E Références
Capteur de température I2C	<u>DS1631</u>
<u>Microcontroleur</u>	PIC16F877
Afficheur LCD	LCD 2×16
LED	LED
Bouton poussoir	BP

TP3 : Mise en œuvre d'un capteur de température I2C.(4h)

Projet n°1 : LED

Projet n°2 : LED_BP

Projet n°3 : LED_BP_LCD

Projet n°4 : LED_BP_LCD_DS1631

Projet n°5 : LED_BP_LCD_DS1631_Thermostat

Projet n°6 : LED_BP_LCD_DS1631_Thermostat_bis

Projet n°1 : LED

Prise en main des logiciels mikroC PRO for PIC (et PROTEUS)

Cahier des charges :

Faire clignoter une LED (ON 1s OFF 1s)(sortie 7 du PORTD)

Prise en main de mikroC PRO for PIC



Steps:	Project type:	
1. Project type 2. Project settings 3. Add files	Standard Create plain project Visual TFT project	
	Select project type that best sutis your needs.	

Choisir Projet standard



Choisir le nom du projet LED, le répertoire de sauvegarde .../LED/, le microcontrôleur 16F877, l'horloge 8Mhz

🏐 Project Setting	s 7 💌
Cim Device	-
Name: P16F877	~
🕸 MCU Clock	
Frequency:	8.000000 MHz
Build / Debugger t	type
Build Type Release	ICD Debug
Debugger Software	i mikroICD

Choisir les paramètres de Project Settings

/* Nom du projet : LED Date de test : Juin 2020 mikroC PRO for PIC v 7.6.0 Q 8Mhz



Pas de fichier à ajouter au projet

C	nikroC PRO for PIC v.7.6.0-C:\Users\thier\Docun
<u>F</u> ile	<u>E</u> dit <u>V</u> iew Project
	5 📴 🖻 诸 📴 🖬 🗋 🖬 🎼
3	👚 Start Page 🖾 📄 LED.c 🜌
Project Settings	1 ⊖void main() {
B	
Code Explorer	

Projet vide

```
*/
void main() {
  PORTD = 0;  // Initialisation PORTD
  TRISD = 0;  // Configuration PORTD en sortie
  while(1) {
    PORTD.F7 = ~PORTD.F7; // toggle D7
    //PORTD = ~PORTD;  // toggle PORTD
    delay_ms(1000);  // tempo 1s
  }
}
```



Copier / Coller du programme ci-dessus LED Compiler le projet "Build", vérifier dans la fenêtre "Messages" il y a des erreurs.

locuments > TP_Capteurs_CESI_2020 > Sujet2020 > LED					
Nom	Modifié le	Туре	Taille		
LED.asm	17/06/2020 11:16	Assembly Language	1 Ko		
LED.bmk	17/06/2020 11:16	Fichier BMK	1 Ko		
LED.brk	17/06/2020 11:16	Fichier BRK	1 Ko		
💼 LED.c	17/06/2020 11:14	C File	1 Ko		
de LED.cfg	17/06/2020 11:16	Configuration Settings	1 Ko		
LED.cp	17/06/2020 11:16	Fichier CP	1 Ko		
LED.dbg	17/06/2020 11:16	Fichier DBG	19 Ko		
LED.dct	17/06/2020 11:16	Recherche gestionnai	6 Ko		
LED.dlt	17/06/2020 11:16	Fichier DLT	1 Ko		
LED.hex	17/06/2020 11:16	Fichier HEX	1 Ko		
LED.log	17/06/2020 11:16	Document texte	2 Ko		
LED.lst	17/06/2020 11:16	Fichier LST	3 Ko		
LED.md	17/06/2020 11:16	Fichier MCL	2 Ko		
ED.mcppi	17/06/2020 11:16	mikroC PRO for PIC	1 Ko		
LED_callertable.txt	17/06/2020 11:16	Document texte	1 Ko		

Regarder maintenant dans le répertoire LED les fichiers qui ont étés crées (en particulier "LED.hex" qui va nous servir pour programmer le microcontrôleur dans PROTEUS)

ᢙ	Start P	age	🖾 📄 LED.c 🔀			\sim	🔲 Watch Value	s			8
	•	1.	Nom du projet : LED			~	i 🗅, 🖻, 💁	🗛 🕹 🕹	I = [🗏 🍡 📣 🔳	
	•		Date de test : Juin 2020				E 📥 Add	🐓 Remove 🔇	Properti	👷 🔛 Add All 🏟	Remove All
	•		mikroC PRO for PIC v 7.6.	0			: - Aug	Keniove 🤇	Flopen	a 🚧 AUU Ali 🎭	Kelliove All
	•		Q 8Mhz				Select variable f	rom list:			
	-	*/	,								~
	• 1	₽v o	id main() {				Search for Varia	bie by assembly h	ame:		
۰	7		PORTD = 0;	// Initialisation PORTD							
۰	•	IT	TRISD = 0;	// Configuration PORTD en sortie							
	• 1	4	while(1) {					11.1			
•	10		PORTD.F7 = ~PORTD.F7;	// toggle D7			Name	value	Addres	5	
			<pre>//PORTD = ~PORTD;</pre>	// toggle PORTD							
•			delay ms(1000);	// tempo 1s							
•			}								
•		ι,									
	-										
						1	1				
							PC= 0x000003	0.00 us			
							Watch Clock				至 🖾
			_			~		Cycles:		Time:	
<					>		Current Count:	0		0.00 us	
***	Messag	ges (🚾 Quick Converter					-			
÷ 🗷	Errors		V Warnings V Hints				Delta:	0		0.00 us	
Line			Message No.	Message Text	U	^	Stopwatch:	0		0.00 us	
0			1	mikroCPIC1618.exe -MSF -DBG -pP16F877 -DL -O11111114 -fo8 -N"C: \Users\thier\Do	ocuments\TP			Reset To Zer	0		
0			1139	Available RAM: 352 [bytes], Available ROM: 8192 [bytes]			Clacks	0	_		
0			122	Compilation Started	L		CIOCK:	°		MHZ	
14			123	Compiled Successfully	U	~					

Faire avancer le programme pas à pas et vérifier le bon fonctionnement en regardant le contenu du PORTD

() Tatrage (⊠ LLD. Band U LD. Charles (LLD. Charl	
1 /4 Nom du projet : LED	- E I . 🏊 I 📖
	- "% 4" 🛄
· Date de test : Juin 2020	Add All 🔗 Remove All
mikroC PRO for PIC v 7.6.0	
Q 8Mhz Select variable from list	
- */	~
void main() { Search for variable by assembly name:	
PORTD = 0; // Initialization PORTD	
• . TRISD = 0; // Configuration PORTD on sortie	
· P while(1) {	
P 10 PORTD.F7 = ~PORTD.F7; // toggle_D7	
//PORTD = PORTD: // toggel PORTD	
♦ 12 dalay ms(1000) // boyse tonib	
PC- 0x00000B 16:00 c	
	3 2
I Watch Clock	÷ 🕹
< Cycles: Tir	me:
Current Count: 32 000 088	6.00 s
111 Messages 📄 Quick Converter	
Verrors Verrings Verlints Delta: 3 1	.50 us
The Measure Net Stopwatch: 32 000 088	6000.04 ms
Lile Presage IV. Presage IV. Deant To Zore	
U 1 mirocupic.bis.exe +nor-bis.phibra/.bit.coll.lllll-ho8-NC:Users/ther/Documents/TP	
0 1139 Available Rum: 552 (bytes), Available Rum: 8192 (bytes) 0 132 Complete State Rum: 8192 (bytes) 133 Complete State Rum: 8192 (bytes) 134 Complete State Rum: 8192 (bytes) 135	Hz
u 122 Compnetori starietu L L 14 123 Completori starietu L L L	
L Compress sources and compres	

Prise en main de PROTEUS

Lancer PROTEUS Professionnel

Dans "Fichier" sélectionner "Nouveau projet"

Choisir Nom : "LED.pdsprj"

Choisir Chemin : " ...\PROTEUS\LED" il faut créer un répertoire "PROTEUS" et un sous répertoire "LED"

"suivant" Choisir Gabarit "DEFAULT"

"Pas de projet firmware" puis "suivant" et "Terminer"





Recherche composants



Recherche dans mots clés "PIC16F877" choisir le microcontrôleur désiré dans résultats et "ok"

Faire la même chose pour rapatrié une LED "LED RED" et une résistance "résistors"

Rapatrier les 3 composants (PIC16F877, LED RED, R) (+ la masse) sur la feuille vierge et réaliser le câblage.



Sélectionner le microcontrôleur 16F877 (en cliquant dessus), puis click droit et choisir "Edit Properties", vérifier l'horloge (8MHz), et choisir dans Program File le fichier à charger dans le micro ici "LED.hex".

Sélectionner le microcontrôleur 16F877 (en cliquant dessus), puis click droit et choisir "Edit Properties", vérifier l'horloge, et choisir dans Program File le fichier à charger dans le micro ici "LED.hex".

Lancer la simulation et vérifier le bon fonctionnement du montage (clignotement de la LED), vérifier le bon timing sur l'oscilloscope en utilisant les curseurs.



Avant de passer au projet suivant rajouter un bouton poussoir de reset "RESET" et arranger le câblage en donnant des noms sur les potentiels utilisés (LED_RD7 et RESET).

Prendre des composants		? ×
Mots clés:	<u>R</u> ésultats (5):	Aperçu BUTTON:
button Tous les mots identiques? Seuls composants modélisés? Catégorie: (Toutes catégories) Microprocessor ICs Switches & Relays	Composant Bibliothèque Description BUTTON ACTIVE SPST Push Button DS1802 MAXIM Dual Audio Taper Potentiometer With Pushbutton Control DS1802S MAXIM Dual Audio Taper Potentiometer With Pushbutton Control DS1802S MAXIM Dual Audio Taper Potentiometer With Pushbutton Control DS1802S MAXIM Dual Audio Taper Potentiometer With Pushbutton Control DS1990 MAXIM Serial Number iButton	Analogue Primitive [RTSWITCH]
Sous-catégorie:		Aperçu circuit: Pet de peckage pour le circuit
Eabricant:		OK Annuler



Tester le RESET en appuyant sur le bouton poussoir "RESET"

Pour éviter d'avoir tout le projet suivant à recâbler Enregistrer le projet LED sous un répertoire ...\PROTEUS\LED_BP avec le nom "LED_BP.pdsprj"

Projet n°2 : LED_BP

Cahier des charges :

Si BP OFF Faire clignoter une LED (ON 1s OFF 1s), si BP ON LED OFF. LED sortie 7 du PORTD et BP entrée 6 du PORTD.

LED_BP sur mikroC PRO for PIC

Créer un nouveau projet LED_BP dans un nouveau répertoire ../LED_BP/

Récupérer le fichier LED_BP.c

Compiler le projet LED_BP

Run et Start Debuger

Dans la fenêtre "watch Values" vérifier le bon fonctionnement du programme en regardant le contenu du PORTD et de TRISD.

Dans la fenêtre "watch Clock" Sur 1 passage dans la boucle vérifier la tempo.



LED_BP sur Proteus

Compléter le câblage avec BP1 relié à la broche 6 du PORTD

Charger le microcontrôleur PIC16F877 avec LED_BP.hex

Lancer la simulation et vérifier le bon fonctionnement du montage :

- que se passe t'il lorsque BP1 est relâché,
- que se passe t'il lorsque BP1 est appuyé,
- que se passe t'il lorsque RESET est appuyé.



Projet n°3 : LED_BP_LCD

Cahier des charges :

Ecrire "votre prénom" sur la première ligne du LCD, écrire "votre nom" sur la deuxième ligne du LCD, si BP OFF Faire clignoter une LED (ON 1s OFF 1s), si BP ON LED OFF. LED sortie 7 du PORTD et BP entrée 6 du PORTD.

LED_BP_LCD sur mikroC PRO for PIC

Créer un nouveau projet LED_BP_LCD dans un répertoire ../LED_BP_LCD/ et récupérer le fichier "LED_BP_LCD.c"

Regarder les routines pilotant le LCD (cliquer sur "Help" ou "?" (bleu)

Rechercher, taper "LCD" comme mot à rechercher et valider

Sélectionner la rubrique "Lcd Library"







Sommaire Index Recherche Entrez le ou les mots à recherche	er Favoris er :			Lcd Library mikroC PRO for PIC Libraries > 1	Hardware Libraries >			0	0	
LCD		~ F								
Afficher let	s nubriques A	fficher								
	, and a construction of the construction of th		Lco	Library						
Sélectionnez une rubrique :	Résulta	sts : 19								
Titre	Emplacement	Rang	The r	nikroC PRO for PIC provides a libr	ary for communication wit	h Lcds (with HD44780 compliant)	controllers) th	ough the	a 4-bit	
SPI Lcd Library	mikroC PRO for PIC	1	Inter	race. An example of Lcd connect	ions is given on the schem	atic at the bottom of this page.				
Lod Library	mikroC PRO for PIC	2	For c	reating a set of custom loc char	acters use Lod Custom Ch	aracter Tool				
SPI Lcd8 Library	mikroC PRO for PIC	3	1010	reading a set of custom Lee char						
mikroC PRO for PIC Comman	mikroC PRO for PIC	4								
Keypad Library	mikroC PRO for PIC	5	Libra	Library Dependency Tree						
OneWire Library	mikroC PRO for PIC	6		,,,,						
Software I2C Library	mikroC PRO for PIC		Icd	I cd Constants						
SPI Graphic Lod Library	mikroC PRO for PIC	8			lits					
Integrated Tools	mikroC PRO for PIC	10								
Manchester Code Library	mikroC PBO for PIC	11								
TET 16-bit Display Library	mikroC PRO for PIC	12	Exte	rnal dependencies of Lcd Li	brary					
TFT Display Library	mikroC PRO for PIC	13	The	following variables must be						
Hardware Libraries	mikroC PRO for PIC	14	def	ined in all projects using I cd	Description :	Example :				
mikroC PRO for PIC Libraries	mikroC PRO for PIC	15	Libr	ary :						
mikroICD Debugger Example	mikroC PRO for PIC	16			Register Releast Free					
Preprocessor Operators	mikroC PRO for PIC	17	exte	ern sir solt <u>DCL_</u> RS:	Register Select line.	solt DCL_RS at RB4_DIT;				
Read Modify Write Problem	mikroC PRO for PIC	18	exte	ern sfr sbit LCD_EN:	Enable line.	<pre>sbit LCD_EN at RB5_bit;</pre>				
Tools	mikroC PRO for PIC	19	exte	ern sfr sbit LCD_D7;	Data 7 line.	<pre>sbit LCD_D7 at RB3_bit;</pre>				
			exte	ern sfr sbit LCD_D6;	Data 6 line.	<pre>sbit LCD_D6 at RB2_bit;</pre>				

Modifier (afin de satisfaire le cahier des charges) et compiler le projet LED_BP_LCD en ayant ajouter au préalable la librairie LCD



LED_BP_LCD sur PROTEUS

Créer un nouveau projet LED_BP_LCD (en partant du dernier projet LED_BP et enregistrer le projet sous)

Récupérer un LCD 2×16 le "LM016L" puis ok

	Prendre des composants			? ×
- -	Mots clés:	Résultats (108):		Aperçu LM016L:
P Devices 9012063A1200FKHFT 3UTTON LED-RED PIC16F877	LCD Tous les mots identiques? Seuls composants modélisés? Catégorie: (Toutos catégories) Ankog 100 Mengrocesser I/Os Optiodectronics TTL 74HC series Sous-catégorie: Eabricant:	Composant Bibliotháq 4054 CMOS 4055 CMOS 74Hc4543 74Hc 74Hc4543 74Hc 74Hc4543 74Hc 74Hc4543 74Hc 74Hc4543 1C4Hc 74Hc4543 1C7Hc 74Hc4740 1DSPLAY FW12A03GLY 1DSPLAY HDG12864F-3 1DSPLAY HDG12864H-6 1DSPLAY HDG12864L-6 1DSPLAY HDM32G512F-3 1DSPLAY HDM32G512F-3 1DSPLAY HDM32G512F-3 1DSPLAY HDM32G512F-3 1DSPLAY HDM32G512F-4 1DSPLAY LDSTK502 1DSPLAY LCDSTK502 1DSPLAY LCDSTK504 1DSPLAY	Description Quad Level Shifters/LCD Drivers With Input Latches BCD To 7-Segment Decoder/Cher Driver BC-Dto-7-Segment Latch0ecoder/Cher for LCDs BC-Dto-7-Segment Latch0ecoder/Cher for LCDs BC-Dto-7-Segment Latch0ecoder/Cher for LCDs I22x32 Graphical LCD with SED1520 controllers 12x434 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1555 controller, Paralel data input 12x434 Graphical LCD with SED1555 controller, Selectable Interface, LED Backlight 12x232 Graphical LCD with SED1550 controllers, Selectable Interface, LED Backlight 12x232 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1555 controller, Selectable Interface, LED Backlight 12x232 Graphical LCD with SED1550 controllers, Selectable Interface, LED Backlight 12x232 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1520 controller 12x434 Graphical LCD with SED1550 controller, Selectable Interface, LED Backlight 12x232 Graphical LCD with Backlight 12x232 Graphical LCD with Backlight 12x244 Graphical LCD with Backlight 12x244 Graphical LCD with SED1520 12x244 12x244 Graphical LCD with Backlight 12x244 Graphical LCD with SED1520 12x244 Graphical LCD with SED1520 12x244 Graphical LCD with Backlight 12x244 Graphical LCD with Backlight 12x244 Graphical LCD with	VSM DEL Model/LEDMEPHA)
		<	>	Annuel

Câbler le LCD avec les directives du fichier "LED_BP_LCD.c" de mikroC PRO for PIC.







Charger le PIC16F877 avec le fichier "LED_BP_LCD.hex"

Lancer la simulation et vérifier le bon fonctionnement du montage



Projet n°4 : LED_BP_LCD_DS1631

Cahier des charges :

Récupérer les informations du capteur I2C DS1631 et afficher la température sur un LCD, si BP OFF Faire clignoter une LED (ON 1s OFF 1s), si BP ON LED OFF. LED sortie 7 du PORTD et BP entrée 6 du PORTD.

LED_BP_LCD_DS1631 sur mikroC PRO for PIC

Créer un nouveau projet LED_BP_LCD_DS1631 dans un répertoire ../LED_BP_LCD_DS1631/

Récupérer le fichier "LED_BP_LCD_DS1631.c"

Analyser ce fichier en se servant de la doc constructeur du capteur DS1631.

Regarder les routines I2C (cliquer sur "Help" ou "?" (bleu)

Rechercher, taper "I2C" comme mot à rechercher et valider



Sélectionner soit "I2C Library" ou "Software I2C Library" suivant les routines utilisées :

Sommaire Index Recherche Entrez le ou les mots à recherche	r Favoris r :	III C Library mikroC PRO for PIC Libraries > Hardware Libraries >
I2C Afficher les Sélectionnez une rubrique :	rubriques Afficher Résultats : 6	I ² C Library
Itre I2C Library Software I2C Library I2C Remappable Library Edit Project Settings	Emplacement Rang mikroC PRO for PIC 1 mikroC PRO for PIC 2 mikroC PRO for PIC 3 mikroC PRO for PIC 4	supports the master MSSP module is available with a number of PIC MCU models. mikroc PRO for PIC provides library which supports the master IPC mode.
Edit Project Settings mikroC PPO for PIC 4 Peripheral Pin Select Library mikroC PPO for PIC 5 STMPE610 Library mikroC PRO for PIC 6	 IPC routines require you to specify the module you want to use. To select the desired I²C, simply change the letter x in the prototype for a number from 1 to 2. Number of I²C modules per MCU differs from chip to chip. Please, read the appropriate datasheet before utilizing this library. 	
		Library Routines 12Cx_Init 12Cx_Start 12Cx_Repeated_Start 12Cx_Is_Jdle 12Cx_Rd 12Cx_Wr 12Cx_Stop 12Cx_Stop 12Cx_SetTimeoutCallback 12C_Set_Active

Dans "library Manager" il faut ajouter les librairies : LCD, I2C, Conversions.

Library Manager			
i 🕏 🐁 🛅 🛅 🖶 🔘			
Search Library Manager			
System Libraries			
🖶 🗌 ADC			
🖶 🗌 Button			
🕀 🗌 CAN_SPI			
Compact_Flash			
Conversions			
⊕ □ C_Math			
🕀 🗌 C Stdlib			
C_String			
B. C_Type			
EEPROM			
EPSON_51D13700			
FLASH			
🕀 🗌 Glcd			
Glcd_Fonts			
🗎 🗹 12C			
🖶 🗌 Keypad4x4			
Ecd Lcd			
Lcd_Constants			
🖮 🗌 Manchester			

Compiler le projet LED_BP_LCD_DS1631

LED_BP_LCD_1631 sur Proteus

Créer le projet LED_BP_LCD_DS1631(en partant du dernier projet LED_BP_LCD et enregistrer le projet sous)

Récupérer le capteur de température I2C le "DS1631" puis ok.



A l'aide de la documentation technique du DS1631, réaliser le câblage avec le microcontrôleur.



Toujours à l'aide de la doc technique :

- Justifier la présence des résistances R4 et R5
- Justifier la mise à la masse de A0, A1, et A2

Charger le PIC16F877 avec le fichier "LED_BP_LCD_DS1631.hex"

Lancer la simulation et vérifier le bon fonctionnement du montage :

• Régler une température de 27°C par ex sur le DS1631 et voir si même chose sur le LCD,

Visualiser sur oscilloscope les signaux SDA et SCL :

- Quel est le niveau des signaux SDA et SCL au repos (mesure avec les curseurs de l'oscillo)
- Isoler la condition de start, la condition de stop
- retrouver l'adresse du DS1631 (lecture),
- retrouver les différentes opérations sur les registres, ... (en se servant du fichier .c)
- Justifier le contenu de chaque registre
- Visualiser les 2 octets de SDA et SCL donnant la température, calculer la température (cf doc)
- Comparer la température avec SDA et SCL, donner la température réglée sur le DS1631, donner la température affichée sur le LCD.



Projet n°5 : LED_BP_LCD_DS1631_Thermostat

Cahier des charges :

Récupérer les informations du capteur I2C DS1631 et afficher la température sur un LCD, programmer les seuils haut TH et bas TL du Thermostat, valider la sortie Tout du DS1631 suivant le cycle choisi, si BP OFF Faire clignoter une LED (ON 1s OFF 1s), si BP ON LED OFF. LED sortie 7 du PORTD et BP entrée 6 du PORTD.

LED_BP_LCD_DS1631_Thermostat sur mikroC PRO for PIC

Créer un nouveau projet LED_BP_LCD_DS1631_Thermostat dans le répertoire .../LED_BP_LCD_DS1631_Thermostat

Récupérer le fichier "LED_BP_LCD_DS1631_Thermostat.c"

Ajouter les librairies nécessaires au programme

Compiler le projet LED_BP_LCD_DS1631_Thermostat

LED_BP_LCD_DS1631_Thermostat sur Proteus

Créer un projet LED_BP_LCD_DS1631_Thermostat (avec enregistrer le projet sous du projet précédent)

Pas de câblage supplémentaire

Programmer le PIC16F877 avec le fichier

"LED_BP_LCD_DS1631_Thermostat.hex"

Lancer la simulation et vérifier le bon fonctionnement du montage :

Visualiser sur oscillo les signaux SDA et SCL et retrouver les différentes opérations sur le bus I2C :

- Condition de start
- adresse écriture, registres de configurations, adresse lecture, info température, ...

Sur deux températures différentes : (avec partie entière et partie décimale)

- donner la température avec 2 octets de SDA et SCL,
- donner la température réglée sur le DS1631,
- donner la température affichée sur le LCD.
- Commentaires

Etude du thermostat :

• Tracer Tout en fonction de la température du DS1631 (on peut simuler la variation de la température en jouant sur le + et le - disponible directement

sur le composant)

• En déduire les seuils TH et TL du thermostat.



Projet n°6 : LED_BP_LCD_DS1631_Thermostat_bis

Cahier des charges :

Récupérer les informations (température, TH, TL) du capteur I2C DS1631 et les afficher sur un LCD :

- Si BP OFF : Afficher la température et Faire clignoter une LED (ON 1s OFF 1s),
- Si BP ON : Afficher TH sur la première ligne du LCD et TL sur la deuxième ligne du LCD et LED OFF.

LED_BP_LCD_DS1631_Thermostat_bis sur mikroC PRO for PIC

A faire ...

• Envoyer par mail le fichier .c

LED_BP_LCD_DS1631_Thermostat_bis sur Proteus

A faire ...

• Envoyer par mail le projet PROTEUS.

Préparation : LED_LCD_DS1621 Norme I2C

Ci-dessous un enregistrement des lignes SDA et SCL dans un exemple de communication entre le PIC16F877 et le DS1621 :



LED_LCD_DS1621.c

39	39 /******Boucle infinie*********************************		
40	while(1)		
41	1 I2C_Start(); //	Détermine si l'I2C est libre et lance le signal	
42	<pre>12C_Wr(0x90); //</pre>	Mode de contrôle en mode écriture	
43	3 I2C_Wr(0xAA); //	Lecture de la température	
44	<pre>12C_Stop(); //</pre>	'Arrêt du sigal	
45	<pre>5 I2C_Start(); //</pre>	Détermine si l'I2C est libre et lance le signal	
46	<pre>6 I2C_Wr(0x91); //</pre>	Mode de contrôle en mode lecture	
47	7 MSB = I2C_Rd(1); //	Nombre signé [température entre +125° et -55°C]	
48	<pre>LSB = I2C_Rd(0); //</pre>	<pre>/Si bit 7 = 1 température MSB +0,5°C</pre>	
49	<pre>9 I2C_Stop(); //</pre>	'Arrêt du sigal	
50	0 if(MSB<0) {		
51	MSB = abs(MSB);	// Si temperature négative valeur absolue de MSB	
52	<pre>2 Temp[0]=45; }</pre>	//Signe - affecté en ascii	
53	3 else {		
54	<pre>Temp[0]=43;)</pre>	//signe + affecté en ascii	
55	5 Temp[1]=(MSB/10)+48;	//Valeur des dizaines affecté en ascii	
56	6 Temp[2]=(MSB%10)+48;	//Valeur des unités affecté en ascii	
57	7 Temp[3]=44;	//Virgule affecté en ascii	
58	8 if (LSB==128) {	//Si bit 7 à 1 alors +0.5*	
59	9 Temp[4]='5'; }		
60	else (//Si bit 7 à 0 alors +0.0°	
61	<pre>1 Temp[4]='0'; }</pre>		
62	<pre>2 Temp[5]=223;</pre>	<pre>// degré affecté en ascii</pre>	
63	3 Temp[6]=67;	//C affecté en ascii	
64	<pre>4 lcd(1, 1, 11, temptxt);//Affiche la chaine de caractère temptxt sur le LCD</pre>		
65	5 delay_ms(100);		
66	<pre>lcd(2, 1, 7, temp); //Affiche la chaine de caractère temp sur le LCD</pre>		
67	delay_ms(100);		
68	s PORTD = ~PORTD; // toggle PORTD		
69	<pre>9 delay_ms(1000);</pre>		
70	0 1		

A l'aide de la boucle infinie ci-jointe et des signaux SDA et SCL ci-dessus :

- Repérer les différents octets présent sur la ligne SDA et en donner leurs significations ?

- Sur cet exemple quelle est la température présente sur le capteur DS1621 ?