



ここではアプリケーションの開発としてターゲット基板モジュールへの部品の取り付け,サンプル・プログラムの実行などを例題として取り上げます.なお,操作例はWindows XPがインストールされたパソコンを使用しています.

## 3-1 アプリケーションの開発準備

 ● ターゲット基板モジュールへ部品追加を行う際の方法(これを行わなくてもサンプル・プログラムの実行はできる) 自分でソフト開発できるように、MSP-FET430開発ツールに付属したヘッダ・ピンを基板に取り付けます.図
 3-1 にターゲット基板モジュールの回路図を示します(水晶振動子は別途入手のこと).

MSP-FET430開発ツールに付属する小物部品は次のように使います.

● 32kHzの水晶振動子の足は写真3-1の位置にあるスルーホールに入れてはんだ付けし、寝かせて取り付ける



**写真 3-1 水晶振動子を取り付けた状態** 〔推奨型番:(株)大真空 DT38(6pF): *f*=32.768kHz〕



写真 3-2 製品に付属するピン・ヘッダ (矢印)を取り付けた状態



図 3-1<sup>(1)</sup> 20/28 ピン用のターゲット基板モジュールの回路図

🕅 mspsampleslac012a.zip	_		
ファイル(E) 編集(E) 表示(V) お気に7	(り(A) ツール(T) ヘルプ(H)		RY
③ 戻る ▼ ● ○ ○ ⑦ ● 検索	∂ 7#11/3 111 -		
アドレス(D) 👔 C¥Documents and Settings	¥ <b>eeee</b> Documents¥ms	psampleslac012a.zip	~ ラ 移動
アドレス(1)       ① マイルをすべて展開         フォルダの作業       ②         ① フィイルをすべて展開       ③         その他       ③         マイドキュメント       ③         サイドキュメント       ③         マイドキュメント       ③         マイドキュメント       ③         マイネットワーク       ○         詳細       ③         mspsampleslac012a.zip       圧縮(5) 形気・フォルダ         更新日時: 2004年9月2日、1309       サイズ: 934 KB	Documents¥ms  i fet120_1.s43  i fet120_1.s43  i fet120_adc10_01_s43  i fet120_adc10_01_s43  i fet120_adc10_05_s43  i fet120_adc10_05_s43  i fet120_adc10_05_s43  i fet120_adc10_05_s43  i fet120_adc10_05_s43  i fet120_adc10_05_s43  i fet120_adc10_01_s43  i fet120_adc10_11_s43  i fet120_adc10_12_s43  i fet120_adc10_12_s43	psampleslac012azip = fet120_fil2s43 = fet120_fil2s43 = fet120_fil2s43 = fet120_fil2s43 = fet120_fil2s43 = fet120_fil2s43 = fet120_roscs43 = fet120_sp0_016xs43 = fet120_sp0_016xs43 = fet120_sp0_01549s43 = fet120_sp0_01estx43 = fet120_sp0_01estx43 = fet120_sp0_01estx43 = fet120_ta_pwm01s43 = fet120_ta_01s43 = fet120_ta_01s43 = fet120_ta_02s43 = fet120_ta_03s43 = fet	
	FET120_demolist_s43.txt	☐ fet120_ta05.s43 ☐ fet120_ta06.s43	

図 3-2 モデル 120 用サンプル・プログラム・ファイルの内容

● 14 ピン・ヘッダはオスとメスがあるので,用途に応じて ZIF ソケットのまわりに立ててはんだ付けする 写真3-2 は,出力ピンの信号をオシロスコープで見ることができるように取り付けた例(状態)です.

●ジャンパ・ピン J4はLED 接続用で、このポートを別用途に使用する場合に取り外せる

J5は電源供給用

ピン・ヘッダを取り付けた基板は,簡単な外付け部品ならばそのままソフトウェア開発に使用できます.デバ イスのすべてのピンが,このピン・ヘッダに出ているので外部との接続に使用できます.

### ■アプリケーション・ソフトウェアの開発の手順

先に述べたとおり、WebにはデバイスごとにMSP430サンプル・プログラムが掲載されています.モデル120 用 Slac012a.zipをダウンロードします.ファイルの内容を図**3-2**に示します.

今回はこれらを使用して,ユーザはこのMSP-FET430開発ツールを使ってどのような手順でソフト開発するのかを説明します.ここでは新規にmsp430testというフォルダを作成してあります.

## 3-2 サンプル・プログラム1手順(温度計測プログラム)

アセンブラ言語のサンプルのソース・プログラムとして,fet120\_adc10\_temp.s43を使うことにします. このプログラムは「msp430F1232」の内部温度センサの電圧を A-D 変換してその温度値をメモリに保存するもの



図3-3 新規ワークスペースを作成して始まる



図3-5 proj1という名前でプロジェクトを保存する



図3-7 この状態でワークスペースを保存する

Create New Project	$\mathbf{X}$
Iool chain: MSP430	<b>▲</b>
Project templates:	
Empty project	
tire asm tire C++	
. E	
Description:	
Creates an emply project.	
	OK Cancel

図3-4 プロジェクトの構成が表示される

💥 IAR Embedded Workbench I	DE
<u>E</u> ile <u>E</u> dit <u>V</u> iew <u>P</u> roject <u>T</u> ools	₩indow <u>H</u> elp
D 🖆 🖬 🕼 🎒 🕹 🕹 🖻	1 ち c
Workspace ×	
Debug 💌	
Files 🔽 🖻	
©proj1 - Deb ✓	
(プロジェクトが)	
啓録される	
prol1	
Ready	

図3-6 左画面に空のプロジェクトが1個登録される



図 3-8 ここでは Souce/Text を選択する

です.メモリには4ビットBCDで結果が入りますので、メモリ内容で温度をそのまま読めます.

この手順例では「msp430test」という空のフォルダを作成し、フォルダ内のワークスペース名は「testwk」、 プロジェクト名は「proj1」、保存するソース・ファイル名は「Adctemp.s43」としています。このサンプル・プ ログラムの実行には、温度センサを内蔵した「msp430F1232」デバイスをターゲット基板モジュールへ取り付け て行います。

手順は次のようになります.

- ●ワークスペース(ここではtestwk)を作成する.
- ●ユーザのソース・プログラムを入力し保存(ここではAdctemp.s43)する.
- ●プロジェクト(ここではproj1)を作成する.
- ●プロジェクトproj1にソース・プログラムを登録する.
- プロジェクトのオプションを設定する.
- リビルドを実行してアセンブルする.
- ●デバッガで実行ファイルを転送する.
- GO コマンドで実行させる.
- (1) キックスタートを起動します.

起動はWindowsの「スタート」メニューから「プログラム」→「IAR Systems→IAR Embedded Workbench KickStart for MSP430 V3」→「IAR Embedded Workbench」と進めます.

最初の画面 (図3-3) で、一番上にある「Create new project in current workspace」のアイコンをクリックしワークスペースを作成します.

(2) [Create new project]  $\forall 47 \text{PD} / (23-4) \text{ ord} = 0 \text{ ord}$ 

ここの例では図**3-5**に示すように,あらかじめ作成してあるmsp430test空のフォルダに,プロジェクト名を 「proj1」とつけて保存したものです.

- (3) 図3-6の左側に空のプロジェクトがproj1として登録されます.
- (4) 次に,新しいWorkspeceを作成します.

Fileメニューの「Save workspace」でWorkspace名を「testwk」として保存します(図3-7).

(5) ソース/ファイルの入力をします.

Fileメニューの「New」にて表示されるダイアログ(図3-8)にて「Souce/Text」を選んで「OK」をクリック します.

(6) 新しい画面が右側に出ますので、ここにアセンブラのソース・プログラムを入力します.

今回はサンプル・コードとして提供されている「fet120\_adc10\_temp.s43」を入力したこととします.この プログラムはチップの内部温度センサにて温度を計測するものです.図3-9に示すように入力して完了させます (この時点ではファイル名はまだUntitledとなったまま).

(7) そのアセンブラ・ソース・テキストはFileメニューの「Save」にて保存します.

このとき,アセンブラ・ソース・テキストの拡張子はs43を使い「ファイルの種類」に注意して保存します. 今回の例では図3-10に示すように,ファイル名を「adctemp.s43」として保存します.



図3-9 ソース・プログラムを入力して完了した状態



図 3-10 ソース・ファイル adctemp.s43 を保存



図 3-11 プロジェクトにソース・ファイル adctemp.s43 を登録



図 3-13 「General Options」の設定

🔏 IAR Embedded Workbench IDE					
<u>File Edit View Project Tools Window</u>	Help				
□ ☞ 🖬 🗿 🎒 🐰 🖻 🛍 🗠 ↔			• > %	¥∓ <u>5</u>	07 0
Workspace × adctemp	.543				
Debug 🔹	mov.w	#1,R10		;	bit t
Files FZ B2 MPY2	bit.w	R10,R11		,	test
	jz	MPY1		;	IF 0:
🗆 🖻 proji - Deb 🗸	add.w	R12,R14		;	IF 1:
Hu 📓 adctemp.s *	addc.w	R13,R15		;	
MPY1	rla.w	R12		;	multi
	rlc.w	R13		;	
(登録される)	rla.w	R10		;	next
	jnc	MP¥2		;	if bi
	ret			;	Retu
				;	
///////////////////////////////////////					
ADC10	_ISR;				
	mov.w	#GIE,0(SP	)	;	Exit
	reti			;	
				;	
;	Intern	upt Vectors	Used MSP4	130x12x	2
,	ORG	OFFFEh		;	MSP43
	DW	RESET		,	
	ORG	OFFEAh		,	ADC10
	DW	ADC10_ISR		,	
	END	-			
proj1					
Beady					T
1 ( ( ( ( ( ( ( ( ( ( ( ( ( ( ( ( ( ( (					

図 3-12 プロジェクト登録された画面(左画面に登録リ ストが表示される)

6.4

Options for node "projl - Debug *" <ul> <li>General Options</li></ul>		A IAK Embedded Workbench IDE	
Options for node "projl - Debue *" <ul> <li></li></ul>		<u>Eile Edit View Project Tools Window H</u> elp	
Options for node "projl - Debue *" <ul> <li></li></ul>			07 Q
Options for node "projl - Debus *"       mov.w #1,R10       2 bit.w         Options for node "projl - Debus *"       mov.w #1,R10       2 bit.w         Files       Popioi - Debus *"       12 mPrl       2 File         General Options       Cotegoor:       Factory Settings       7 mov.w       7 mov.w         Cotegoor:       Factory Settings       7 mov.w       7 mov.w       7 mov.w         Cotegoor:       Factory Settings       7 mov.w       7 mov.w       7 mov.w         Cotegoor:       Factory Settings       7 mov.w       7 mov.w       7 mov.w         Cotegoor:       Factory Settings       7 mov.w       7 mov.w       7 mov.w         Setup Incore       Factory Settings       7 mov.w       7 mov.w       7 mov.w         File Debugger       Setup Incore       7 mov.w       7 mov.w       7 mov.w         Setup Incore       Factory Tice       7 mov.w       7 mov.w       7 mov.w         File Debugger       Setup Incore       7 mov.w       7 mov.w       7 mov.w         File Debugger       File Debugger       7 mov.w       7 mov.w       7 mov.w         File Debugger       File Debugger       7 mov.w       7 mov.w       7 mov.w		Workspace × adctemp.s43	
* Messages Device description file Override default FTOOLLIT_DIRSHconfielMap400F1222.ddf  * Messages  D bytes of CODE memory (+ 174 absolute ) Errors: none Warnings: none  * Messages  * Message	Calegory:       Factory Settings         Citer complet       Setup   Plueins           Cuttom Build       If Endourcer         Part Bebagging       Setup macro         Simulator       General Options         Obdogging       Setup in plueins           Obdogging       Setup in plueins           Device description file       Device description file         Override default       If Device description file	Viol source     Debug     Debug       Files     レ       Debug     Improvide       Debug	bit test IF 0. IF 1. mult: Return
Total number of errors: 0		Total number of errors: 0 Total number of warnings: 0	
OK Cancel Ready	OK Cancel	C Ready	

図 3-14 ドライバは「FET Debugger」を選択

図3-15 アセンブル結果画面

(8) 次に、今作成したソース・プログラム・ファイルを Proj1 に登録します.

Project メニューの「Add File」にて登録します. 図3-11 に示すとおり「ファイルの種類」をアセンブラにして,登録するファイルの「adctemp.s43」を指定して「開く」ボタンをクリックします.

(9) 登録された画面を図3-12に示します.

(10) プロジェクト"proj1"のオプションの設定をします. Projectメニューの「Options」で表示される画面(図
 3-13)の「Category」にある「General Options」,「Target」タブでは次の2箇所を設定します.

●「Device」プルダウン・メニューでこれから使用する「msp430F1232」を選択

● [Assembler only Project] ヘチェック

(11) 引き続き「Category」を「Debugger」に変え、変わった画面の「setup」タブの「Driver」プルダウン・メニューでは、図3-14で示すようにパソコンに接続したMSP-FET430開発ツールHWの「FET Debugger」を選びます。

その他のカテゴリでは、とくに設定は不要です.それぞれ右上の「factory Settings」ボタンをクリックす ると初期設定になります.最後に「OK」ボタンをクリックしてプロジェクトのオプション設定を終わります.

(12) では、ソース・ファイルadctemp.s43をアセンブルしてみましょう. Projectメニューの「Rebuiled All」
 をクリックするとアセンブルされます. アセンブル結果は図3-15のように下側のウィンドウに表示されます. もし、エラーが表示されたらソース・テキストを修正し再度アセンブルしエラーが表示されないようにします.

(13) ターゲット基板モジュールのデバイスにこのプログラムを転送するためにProjectメニューの「Debug」を

クリックします.プログラムが転送され、図3-16に示すように右にもう一つのウィンドウが表示されます.

(14) まず, debugメニューの「Go」で実行させ, すぐ debugメニューの「Break」で停止させます.

ightarrow IAR Embedded Workbench	IDE			
<u>File Edit View Project Debug</u>	Emulator Tools Window	v <u>H</u> elp		
D 😅 🖬 🕼 🖉 🐰 🖻 🖻	L D CI	▼ > > ½ ∑ □ □ □ □ □	9	
ちしちゃんちちち	*   ろ   〇			
			• Disassembly	×
Debug	adctemp.s43		Goto	
	; February 2004		A DECET.	
	/ #include "msp430x:	12x2.h"	⇒ 00E000	31400003
E Sproji - Deb V	;		StopWDT StopWDT	nov.v
Here Carlo Contract	ORG	OE000h ; Program 5	00E004	B240805A2
	⇒RESET BOY.W	#0300h.SP ; Initializ	SetupADC1 SetupADC1	0:
	StopWDT mov.w	#WDTPW+WDTHOLD,&WDTCTL ; Stop WDI	OOEOOA	B24060A0E
	SetupADC10 mov.w	<pre>#INCH_10+ADC10DIV_3,&amp;ADC10CTL1</pre>	00E010	B2403838E
	nov.w	#SREF_1+ADC10SHT_3+REF0N+ADC100N+AI	Mainloop	bis.v 📟
	Mainloon <b>bis w</b>	#ENC+ADCIOSC &ADCIOCTLO : Start say	Mainloop: 00E016	B2D00300E 🗸
proj1	fo 4		<	>
* Messages				
Building configuration: proi1 -	Debua			
Configuration is up-to-date.				
Debug Log Build				×
reauy				

図3-16 ターゲット基板モジュールにプログラム転送完了画面

🗶 IAR Embedded Workbench 🛛	IDE				
<u>Eile Edit View Project Debug</u>	; Emulator <u>T</u> ools <u>W</u> indow	Help			
D 🖨 🖬 🕼 🐰 🖪 🖻	1 0 0 J	<ul> <li>▼ № № №</li> </ul>	7   Di 😲 🕅   4	3	
5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5	7 Z Ø				
125 an at l ⊕ ⊗					
Workspace ×	adctemp.s43		+ ×	Disassembly	×
Debug 💌	Trans2TempC;Subrout	tine coverts R12 = ADC10	MEM/1024+423	Go to	• M
Files 🔽 📴	; oC = ((	(x/1024)*1500mV)-986mV)*	1/3.55mV = x	0001EA	FFFF0000 🔨
🗉 🖻 proj1 - Deb 🗸	; Input:	ADC10MEM 0000 - 0FFFh	, R11, R12, .	0001EE 0001F2	FFFF0000
dctemp.s	;			0001F6 0001FA	FFFF0000
Uutput	⊂> mov.w	«ADC10MEM,R12	;	0001FE	FFF 330
	mov.w	#423,R11 #MPVII	; c	000202	80701E00
	bic.w	#00FFh,R14	; /1024	000208	E85B0128
II	add.w	R15,R14	2	00020E	3A17
	swpb rra.w	R14 R14	· ·	000212	9324
proj1	fo I			<	
* Messages					
Changed settings forces a full	l rebuild				
Rebuilding configuration: proj	1 - Debug				
C\Documents and Settings\n C\Documents and Settings\n	ewsate\My Documents\ms	sp430test\Debug\Obj\adcter	np.r43 449		(温度)
C. (Documenta data Detanga (	ewadle phy Documenta pha	phanes (pepagitexelpio)	415		
2 file(s) deleted.					
adatama a 42					
auciempisaj	.d Cotting of non-solution of the	normantal man 1904a at Dahur			
E Debug Log Build					×
Ready					

図 3-17 現在の温度を計測するため実行した画面(この例では 33℃を表示)

右側の disassembly ウィンドウをスクロールさせ,図3-17 のように 200 番地が見える位置まで移動します.200 番地には 33 が,202 番地は 91 を表示しているのがわかります.これはこのプログラムが内部温度センサで検出したものを A-D 変換して 200 番地と 202 番地にストアしています.そこで直接メモリの内容を見ると,現在の温度

XIAR Embedded Workbench	IDE						
Eile Edit View Project Debug	g Emulator <u>T</u> ook	: <u>W</u> indow	Help				
				- <u>-</u>	07 109 08 4	9	
	+ X 42				- 10 <b>- 410</b> - 10   20		
	7 27 88						
∑ ∝ 数  0, 0 □□  ● ∛	*						
Workspace ×	adctemp.s43				+ ×	Disassembly	×
Debug 💌		Output:	R15, R14			Goto	▼ M
Files 🔽 🗠	;				<b>_</b>	0001EA	FFFF0000 🔨
R Paroil - Deb		clr.w	R14	;	0 -> LSB:	0001EE	FFFF0000
La Badctomp c		clr.w	R15	;	0 -> MSB:	0001F2	FFFF0000
La Contract	MACU	clr.w	R13	;	MSBs mult	0001FA	FFFF0000
a a oupur		MOV.W	#1,R10	;	bit test	0001FE	EEE 3400
	5>MPY2	bit.w	R10,R11	;	test acti	000202	9300
		jz	MPY1	;	IF 0: do	000208 /	E85B0128
		add.w	R12,R14	;	IF 1: add	00020C	400B
		addc.w	R13,R15	;	_	00020E	3À17 /
	MPY1	rla.w	R12	;	multiplie	000210	4864
		rlc.w	R13	, , ,	-	000212	9324
proji	<u>lfo</u> l <b>∢</b>				•	<	
* Messages							
Building configuration: proj1 -	Debug						
Configuration is up-to-date							
configuration is up-to-date.							
-							
Debug Log Build							×
Ready							
r waay							

図3-18 しばらく動かしたときの温度計測画面(この例では34℃を表示)

IAR Embedded ¥	orkbench IDE		$\mathbf{X}$
Save pr	oject "proj1"?		
THE REAL PROPERTY OF THE PROPE	いいえ(N)	キャンセル	

図3-19 終了時のプロジェクト保存画面

として摂氏33度と華氏91度を見ることができます.数字(温度)は実際に実行している場所での温度ですので, 必ずしもこの表示と同じ値にはなりません.

(15) 次は, デバイスの内部温度が上がるようにして debug メニューの「Go」で再び実行させ, しばらくそのま まにします.

温度が上がった頃に debug メニューの「Break」で停止させます. 同様に 200 番地と 202 番地を見ると数字が 変わって温度が変わっているはずです. この例では,図3-18 に示すとおり 200 番地で 34 度まで上がった状態が観 測できました.

(16) 何度か試したら、メニューのdebugメニューの「Stop Debugging」でデバッガを停止させます.

(17) これでテストが終わったのでキックスタートをFileメニューの「Exit」で終了します.今回はプロジェクト変更したので,図3-19に示すようにメッセージが出ますのでプロジェクトを保存ください.

## 3-3 サンプル・プログラム1 (温度計測プログラム)の説明

サンプル・プログラムを理解するに有効なドキュメントはソース・プログラムのコメント(英文)と特に 「MSP430x1xx Familyユーザーズ・ガイド(SLAU049)」の日本語版があります.このユーザーズ・ガイドにはデ バイスの機能解説と命令語の説明があります.

ここで使用したデバイスには内蔵温度センサがありますので、A-D変換後の電圧値から温度を計測するプログラムを作ることができます。内蔵温度センサが出力する電圧は10ビットの値でメモリのADC10MEMに入ります。

メイン・ループは次のリストのように、A-D変換開始後は変換完了割り込みが起こるまで命令を停止させて待っています.完了すると次の命令に進み、摂氏と華氏温度4ビットBCD形式に変換しメモリへストアします.

Mainloop bis.w #ENC+ADC10SC, & ADC10CTL0 ; A-D変換を開始させる bis.w #CPUOFF+GIE, SR ; LPM0モードで待機,割り込みが発生したら次の ; 命令へ ; A-D変換結果がメモリ ADC10MEMに入る

メモリの ADC10MEM の値を OUT とすると摂氏温度は次の式で計算できます.

温度 =  $\left(\frac{\text{OUT} \times 1.5}{1024} - 0.986\right) \times \frac{1}{0.00355} = \frac{\text{OUT} \times 423}{1024} - 278$ 

# 3-4 サンプル・プログラム2手順(PWM波形出力)

次に,もう一つの例を示します.同じ「msp430test」というフォルダにPWM波形を出力するソフト開発フ ァイルの作成例を示します.workspace名は先と同じ「testwk」,プロジェクト名は別の「proj2」,アセンブラ 言語のサンプルのソース・プログラムは「fet120 ta pwm01.s43」を使うことにします.

作成はサンプル・プログラム1の手順と同様ですが.プロジェクトの追加とソース・プログラムの追加する部 分が異なります.

PWM 波形を出力するこのプログラムを簡単に説明します. これは約 800kHz のクロックを使い,周期約 640 μs でデューティ比 75%の波形をポート P1.2 へ,デューティ比 25%をポート P1.3 に低消費電力モードで連続出力します. この波形の確認にはオシロスコープなどの測定器を使います.

このサンプル・プログラム実行にはほかのデバイスでもかまいません. この例ではターゲット基板モジュールへ 「msp430F1232」を取り付けて行っています.

(1) キックスタートを起動します.

最初の画面 (図3-20) で「Open existing workspace」のアイコンをクリックしワークスペースを開きます. すでにキックスタート起動中の場合はファイル・メニューから Workspace を開きます.

(2)「Open existing workspace」ダイアログ(図3-21)では先に作成した「testwk.eww」を選び「開く」 をクリックします.



図3-20 既存のワークスペースを選択する

A IAR Embedded Workbench IDE	
Eile Edit View Project Tools Window Help	
□ ■ ■ ● ※ ● ■ ● > > ■ ▼ ★ ★ 国 時 曜 検	5
Workspace x adctemp.s43	
Debug     Image: Constraint of the second seco	Create New Project
ped fin i	Description: Creates an empty project. DK Cancel

図 3-22 既存ワークスペース testwk.eww を開いた画面

```
図 3-23 Create New Project 画面
```

(3) 図3-22のように左側に過去に登録した内容が表示されたりします.

今回はプロジェクト名proj1が表示され、右側にはproj1のソース・プログラムが表示されています.前回実 行した内容が表示されますので、必ずそのとおりにならないことがあります。

(4) 次に,新しいプロジェクト名「proj2」を追加します.

projectメニューの「Create New Project」で表示される画面(図 3-23)では「OK」をクリックします. こ



図 3-24 プロジェクト proj2 保存画面



図 3-25 Souce/Textを選択



図 3-26 PWM 波形出力ソース・プログラムのテキスト 入力完了画面

図 3-27 ファイル名をpwm.s43 で保存

こでは、図**3-24**に示すようにmsp430testフォルダにプロジェクト名を「proj2」とつけて保存します. (5) ソース・テキストの入力を行います.

Fileメニューの「New」をクリックし、表示されるダイアログ(図3-25)では「Souce/Text」を選んで「OK」 をクリックします.

(6) ソース・テキストの入力のための新しい画面が右側に出ますので,ここにアセンブラのソース・プログラム を入力します.今回はサンプル・コードとして提供されている「fet120\_ta\_pwm01.s43」を入力したこととし ます.このプログラムはPWM波形をポートへ出力するものです.図3-26は入力が完了した状態です.

(7)入力が完了したアセンブラ・ソース・テキストはFileメニューの「Save」にて保存します.

このときアセンブラ・ソース・テキストの拡張子には s43 を使います. 今回は図 3-27 に示すようにファイル名 を「pwm.s43」で保存します.

(8) 次にソース・プログラム・ファイルをプロジェクトproj2に登録します.
 projectメニューの「Add File」にて登録します.図3-28に示すとおり「ファイルの種類」をアセンブラにし

Add Files - proj2	ļ				?×
ファイルの場所の	🗁 msp430test		• +	• 🖬 🔁	
最近使った77イル	Debug Settings adctemp.s43				
デスカトップ	⊡[pem.s43				
7/ 1423/2/1					
₹1 I)/II-9					
₹1 x9F9-0	7=(1.4.00	Trans = 43			RB( (D)
	ファイルの種類①	Assembler Files (*.s*,*.n	nsa:*.asm)	-	キャンセル

図 3-28 ソース・ファイルをプロジェクト Proj2 へ登録

Options for node "proj1 - Debug \*"



図3-29 プロジェクトに登録された画面



図 3-30 「General Options」の設定



て,登録するファイル「pwm.s43」指定して「開く」ボタンをクリックします.

(9) 登録された画面を図3-29に示します. 左側のウィンドウにはproj1,proj2が,右ウィンドウにはソース・ テキストが合計2件入っていることがわかります.

(10) 先にproj1と同様にプロジェクトproj2のオプションの設定をします. Projectメニューの「Options」 で表示される画面(図3-30)では「Category」での「General OPtions」, targetタブでは次の2ヵ所を設定 します.

●「Device」プルダウン・メニューではこれから使用する「msp430F1232」を選択

●「Assembler only project」へチェックを入れる

(11) 引き続きCategoryを「Debugger」に変え, setupタブの「Driver」では図3-31のように「FET Debugger」を選びます.

その他のカテゴリでは特に設定は不要です. それぞれ右上の「factory Settings」ボタンをクリックすると









初期設定になります.最後に「OK」ボタンをクリックしてプロジェクトのオプション設定を終わります.

(12) 終わったらアセンブルします.

Project メニューの「Rebuiled All」を実行すると「pwm.s43」がアセンブルされます. 結果は図3-32のように下のウィンドウに結果が表示されます. エラーが表示されたらソース・テキストを修正し再度アセンブルします.

(13) 基板にプログラムを転送するために、Projectメニューの「Debug」をクリックします.

プログラムが転送され、図3-33のように右にもう一つのウィンドウが表示されます.

(14) debugメニューの「Go」で実行させます. PWM出力は基板の24番ピンのポートP1.3に25%デューティ比の波形が,23番ピンのポートP1.2には75%の波形が出ます.出力をオシロスコープで波形を観測したものが写真 3-3です.

(15) 観測が終わったら debug メニューの「Stop Debugging」でデバッガを停止させます.

🔏 IAR Embedded Warkbench	IDE				
Eile Edit View Broject Debug	t Emulator <u>T</u> ools <u>W</u> indo	n Help			
D 📽 🖬 🖉 🚳 X 🖻 🖻	8 10 CX	<ul> <li>&gt; &gt; &gt; &gt;</li> </ul>	0   03 😭 Pš   3	8	
5 - 525532	5 A 9				
X = a a a a a a a					
Workspace	Fire man. The stratistic		- ×	Dearsenbly	
Debug	·********			Goto	•
Files 🔽 🗠	finclude "msp430x	12x.h"	<u> </u>	RESET :	
B Sproj2 - Deb	;	07030h	7 Program	StopVDT:	31400003 hov.v
0verview pog1 pog2	© RESET         nov.w           StopADT         nov.w           SetupP1         bis.b           BetupC1         nov.w           SetupC1         nov.w           Model         nov.w           SetupC1         nov.w           Model         nov.w	#300h.3P #UT794-#UTHOLD, ANDTOTL #00Ch.4PLDIR #00Ch.4PLSEL #512-1,4CCR0 #0UTHOL_7,4CCTL1 #384,4CCEL	<pre>&gt; Tmitiali &gt; Stop WOZ &gt; 21.2 and &gt; 21.2 and &gt; 20.2 and &gt; 20.2 and &gt; 20.2 and &gt; 20.2 pays &gt; 0.021 pays &gt;</pre>	SetupP1 SetupP1: 00F00A 00F010 SetupC0 SetupC0: 00F016	bis.b F2D00C002 F2D00C002 B240FF017
Log Sat Sep 25 14 16:37 2004: Inte Set Sep 25 14 16:38 2004: De Sat Sep 25 14 16:41 2004: Los Set Sep 25 14 16:41 2004: Ta Set Sep 25 14 16:42 2004: Der	artisce di version 2.10 vice : MSP430F12x2/F15 aded module rget reset ver does not support code	2 e coverage.			
Debug Log Build					
Ready					

図 3-33 転送完了画面

IAR Embedded Workbench IDE					
⚠	Save proj	ject "proj2"?			
(#C)	w_1	いいえ(N)	キャンセル		

図3-34 終了時のプロジェクト保存画面

File メニューの「Exit」で終了します. このとき図 **3-34** に示すようにプロジェクトを保存します. また work space の保存が出たときも保存します.

## 3-5 サンプル・プログラム2 (PWM波形出力)の説明

PWM 波形の出力はここで使用した下記のソース・リストで示すとおり,非常に簡単にできます.

#include	"msp430x1	2x.h"	
;			
RESET	mov.w	#300h,SP	; Initialize 'x12x (2) stackpointer
StopWDT	mov.w	#WDTPW+WDTHOLD,&WDTCTL	; Stop WDT
SetupP1	bis.b	#00Ch,&P1DIR	; P1.2 and P1.3 output
	bis.b	#00Ch,&P1SEL	; P1.2 and P1.3 TA1/2 otions
SetupC0	mov.w	#512-1,&CCR0	; PWM Period
SetupC1	mov.w	#OUTMOD_7,&CCTL1	; CCR1 reset/set
	mov.w	#384,&CCR1	; CCR1 PWM Duty Cycle
SetupC2	mov.w	#OUTMOD_7,&CCTL2	; CCR2 reset/set
	mov.w	#128,&CCR2	; CCR2 PWM duty cycle
SetupTA	mov.w	<pre>#TASSEL_2+MC_1,&amp;TACTL</pre>	; SMCLK, upmode
			;
Mainloop	bis.w	#CPUOFF,SR	; CPU off
	nop		; Required only for debugger
			;

CCR0のタイマ・カウントはPWM全体の周期(512\*1/800,000)を, CCR1のカウント値は384/512=75%のデュ ーティ比のパルス幅, CCR2のカウント値は128/512=25%のデューティ比のパルス幅を繰り返しで出力します. この値を変更することにより,希望の周期やデューティ比のパルスを出力できます(図3-35).このポートへの出



図3-35 各パルスの幅

SIAR Embedded Workbench	IDE					
Eile Edit View Broject Debug	Emulator Tools Window Help					
0 🖨 🖬 🕼 🖓 🖓 👘 🖻	<u>.</u>   12 CH   .	* # ゴ 国   時 端 ※   効				
5 6 826888	- B 🛱					
2 ~ 2 0 0 0 0 0 0						
Workspace ×	fet120 1.s43 fet120 1.c					
Parallel port Debug 🔹	// December 2003					
Files 🔽 🛱	// Updated for IRR Embedded WR	orkbench Version: 2.21B				
⊡ 15 fet120 1 C ✓	//*************************************					
Heal Difet120_1.c	#include <msp430x12x.hb< td=""><td></td></msp430x12x.hb<>					
	woid main(woid)					
	· · · · · · · · · · · · · · · · · · ·					
	PIDTE Is DrD1:	// Set FL-0 to output direction				
	for (33)					
	unsigned int i;					
(アクティブにす	5 PLOUT ~= 0x01;	// Toggle \$1.0 using exclusive-OR				
( ) ) ) ) ( )	1 - 50000;	// Delay				
	do (1);	,,,,				
	while (i != 0);					
$\frown$	1 . '					
ler120_1_Ciet140_1_asm _ + +	Ifo I					
Building configuration tet120	1 C - Parallel port Debug					
containing coningurations names,	, _o r a antipart babog					
Configuration is up-to-date.	トコン	パイル結果				
	)					
Debug Log Build						
land.						

図3-36 C言語で作られたLED 点滅サンプル・プログラムのコンパイル結果

力は,

Mainloop bis.w #CPUOFF,SR

のコーディングのところで止まったまま低消費電力モードに移行して持続します.

PWM出力は基板に取り付けたヘッダ・ピンの23番ピンまたは24番ピンにオシロスコープなど接続してポート からの出力波形を見ます.これでポートP1.3に25%デューティ比の波形が,ポートP1.2には75%デューティ比の 波形が連続で出ていることが確認できます.ピン番号は各デバイスのデータ・シートに記載されています.

## 3-6 LED点滅サンプル・プログラムのC言語版の例

MSP-FET430 開発ツールにはCコンパイラ (コード・サイズ 4K バイトまでの使用制限あり) が付属しています ので、C言語で書かれたプログラムも試してみます.先に第2章2-4で実行したLED 点滅サンプル・プログラムの C言語版がその下にサンプルとして入っていますので、これを動かす操作例を説明します.

実行したLED 点滅のアセンブラ・プログラム画面で、左下のタブをクリックしてC言語のサンプル・プログラムをアクティブにします.これをアセンブラと同様にリビルドしてデバッガを動作させ、基板側に転送して実行すれば同様な動作になります.

図3-36に示すのは、C言語で書かれたサンプル・プログラムを選択し、リビルドした時点の画面です.

#### ●引用文献●

(1) MSP-FET430ユーザーズガイド, p.B-6, テキサス・インスツルメンツ.