第3章　アプリケーションの開発例

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

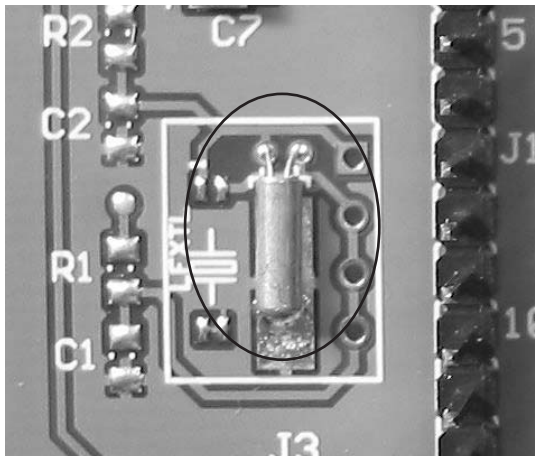

写真3-1　水晶振動子を取り付けた状態
〔推奨型番：(株)大真空　DT38(6pF)：f=32.768kHz〕

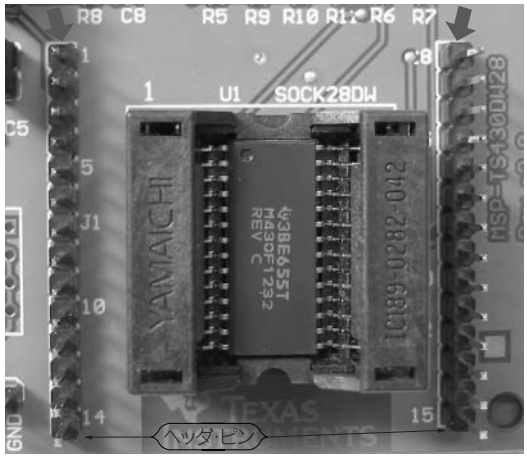

写真3-2　製品に付属するピン・ヘッダ(矢印)を取り付けた状態

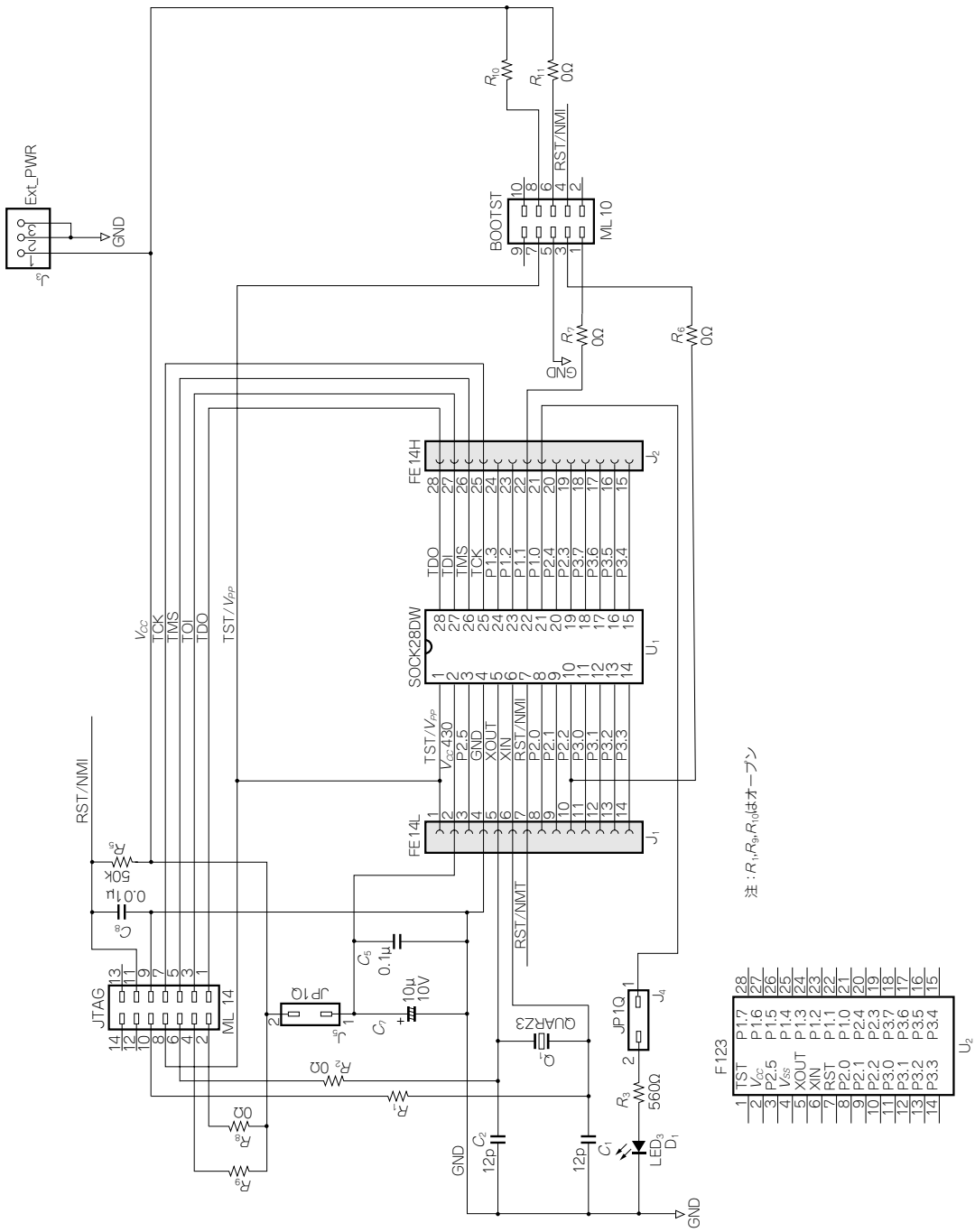

図3-1(1) 20/28ピン用のターゲット基板モジュールの回路図

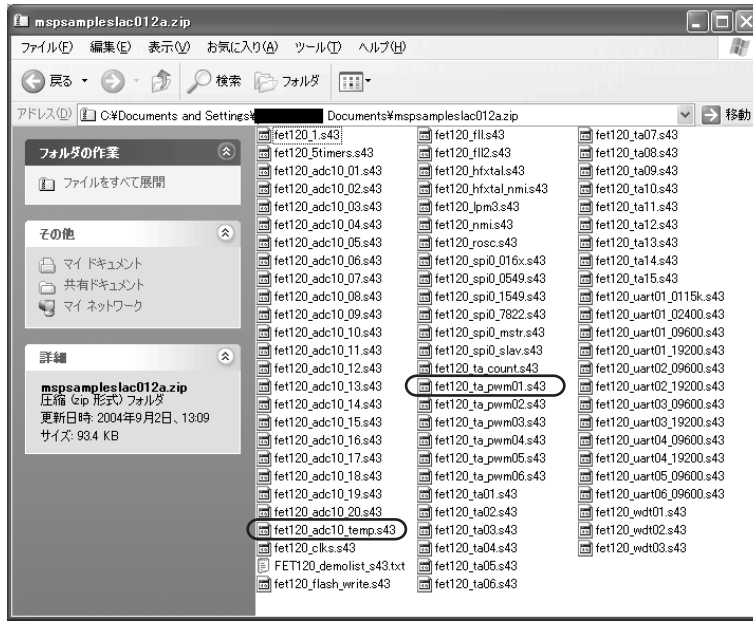

図3-2　モデル120用サンプル・プログラム・ファイルの内容

● 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

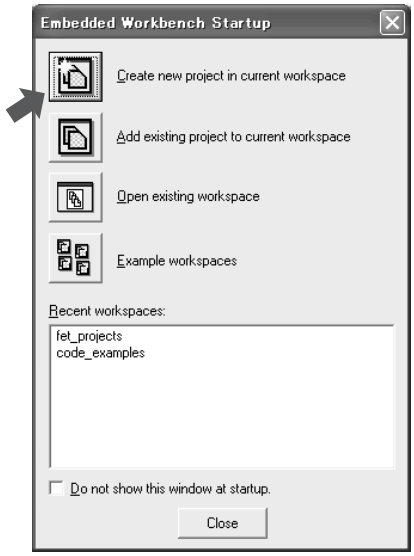

図3-3　新規ワークスペースを作成して始まる

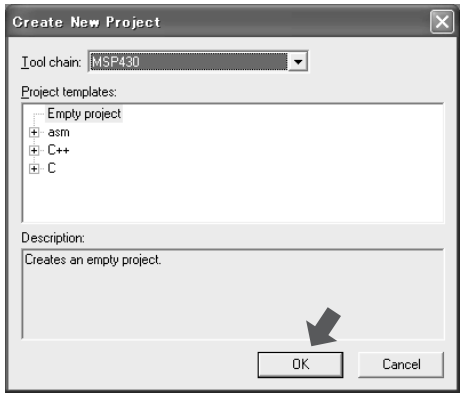

図3-4　プロジェクトの構成が表示される

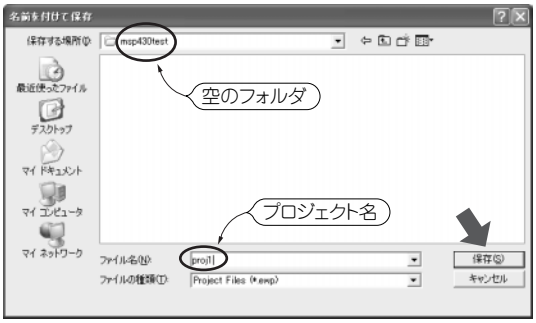

図3-5　proj1 という名前でプロジェクトを保存する

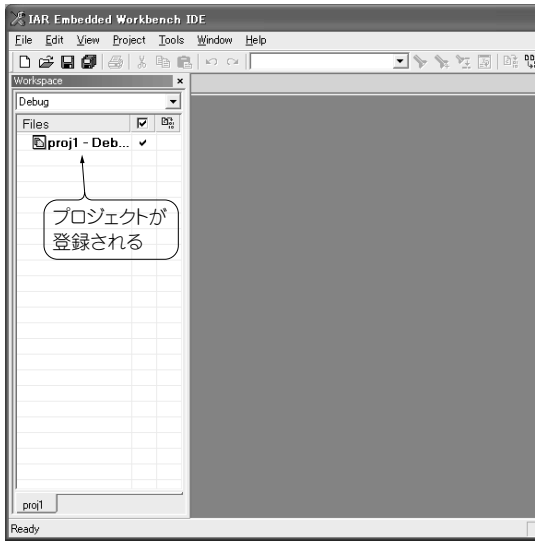

図3-6　左画面に空のプロジェクトが1個登録される

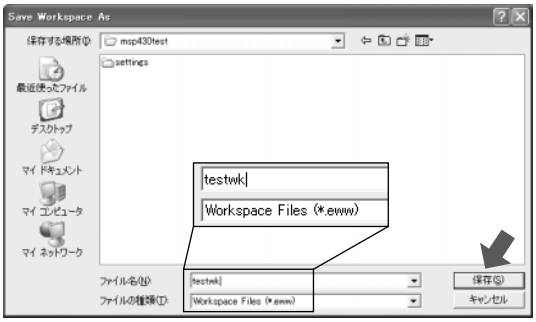

図3-7　この状態でワークスペースを保存する

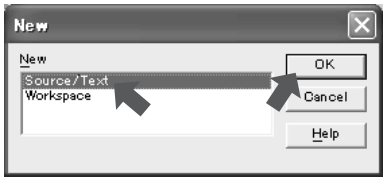

図3-8　ここでは Souce/Text を選択する

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」ダイアログ（**図3-4**）ではそのまま「OK」をクリックします．

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．

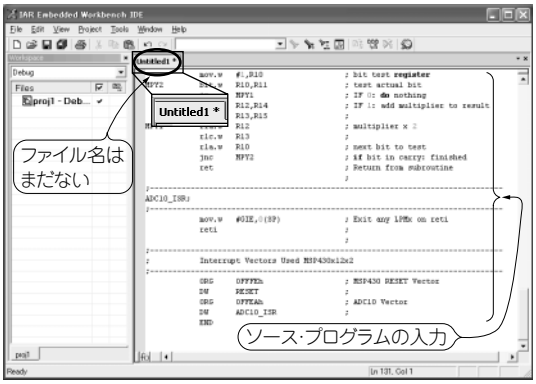

図3-9　ソース・プログラムを入力して完了した状態

図3-10　ソース・ファイルadctemp.s43を保存

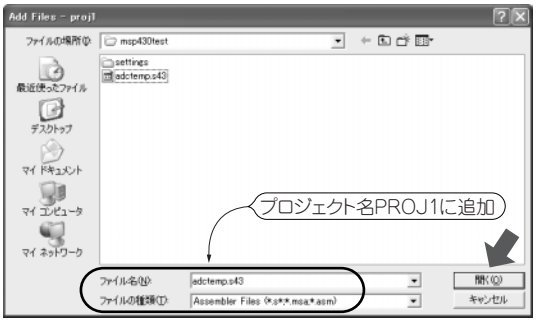

図3-11　プロジェクトにソース・ファイルadctemp.s43を登録

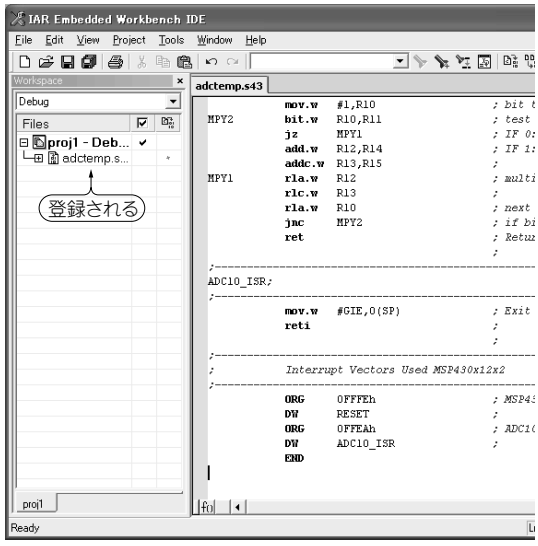

図3-12　プロジェクト登録された画面（左画面に登録リストが表示される）

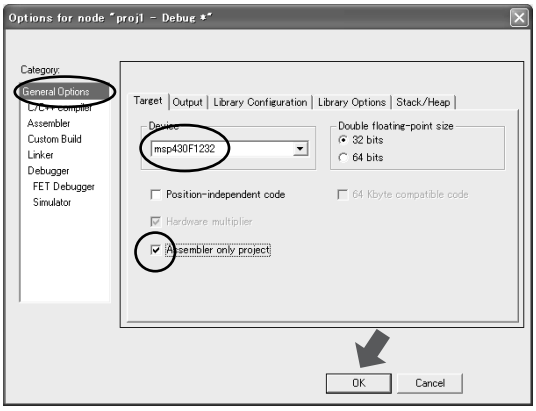

図3-13　「General Options」の設定

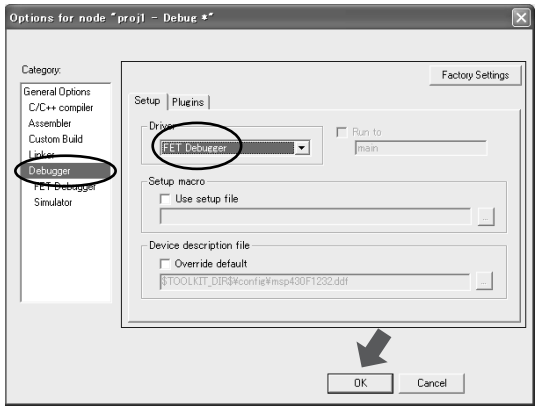

図3-14　ドライバは「FET Debugger」を選択

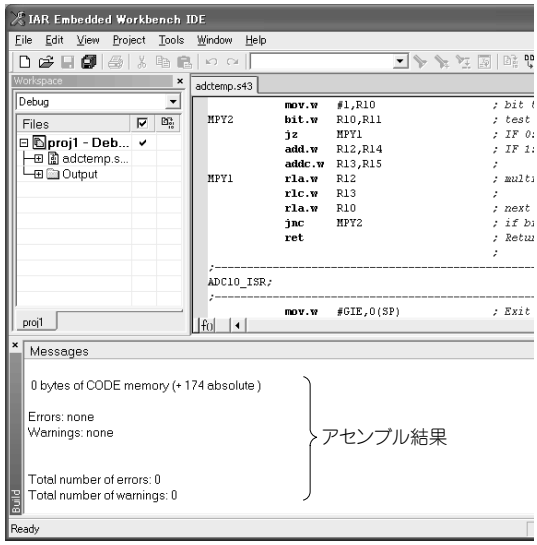

図3-15　アセンブル結果画面

(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」へチェック

(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．

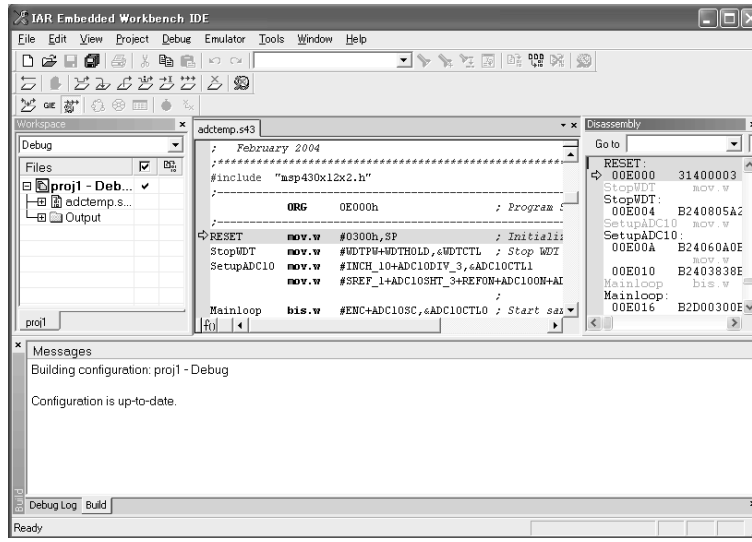

図3-16　ターゲット基板モジュールにプログラム転送完了画面

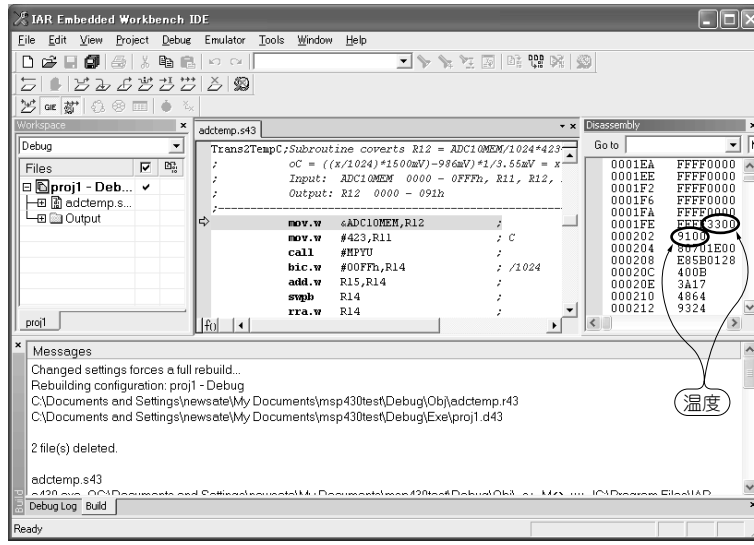

図3-17　現在の温度を計測するため実行した画面（この例では33℃を表示）

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

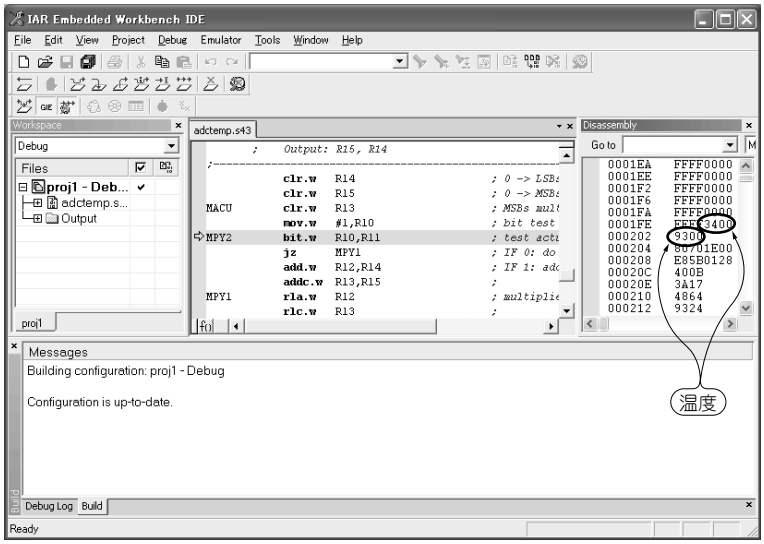

図3-18　しばらく動かしたときの温度計測画面（この例では34℃を表示）

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．

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

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．

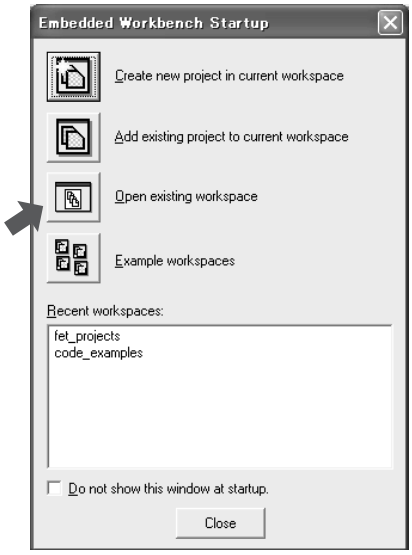

図3-20　既存のワークスペースを選択する

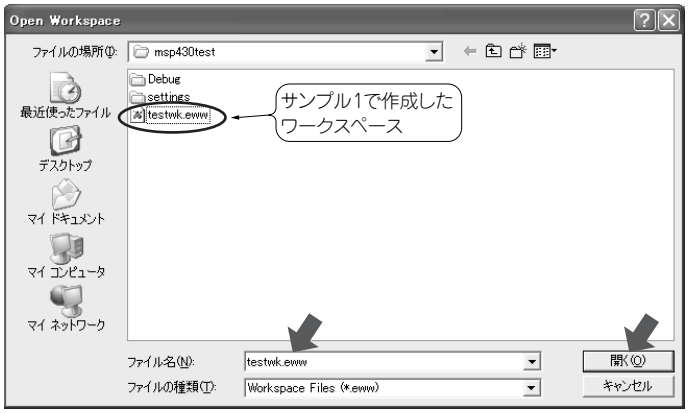

図3-21　既存ワークスペース選択

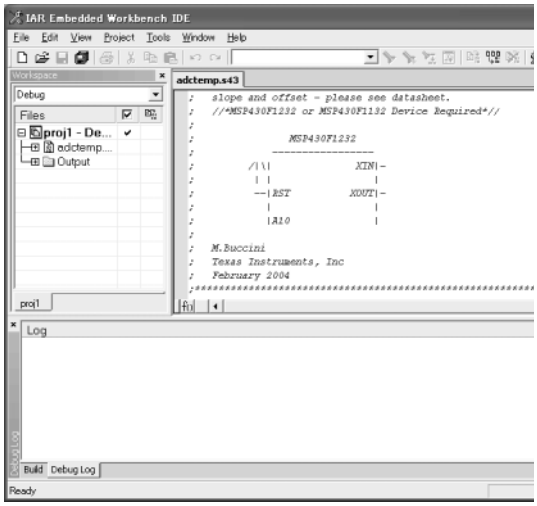

図3-22　既存ワークスペースtestwk.ewwを開いた画面

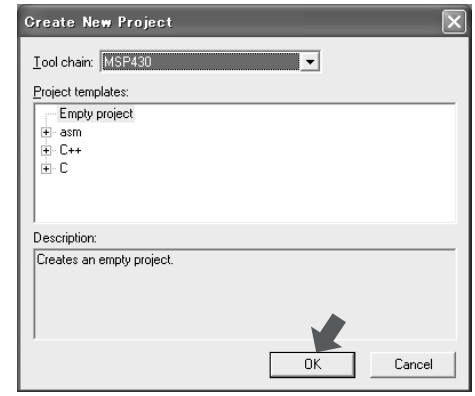

図3-23　Create New Project画面

(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

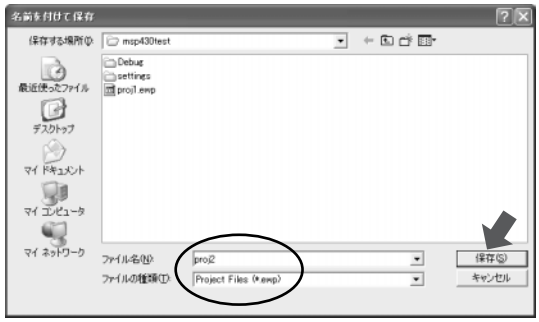

図3-24　プロジェクト proj2 保存画面

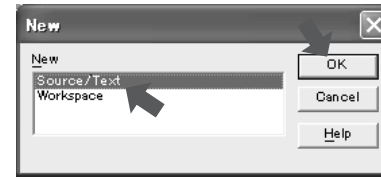

図3-25　Souce/Text を選択

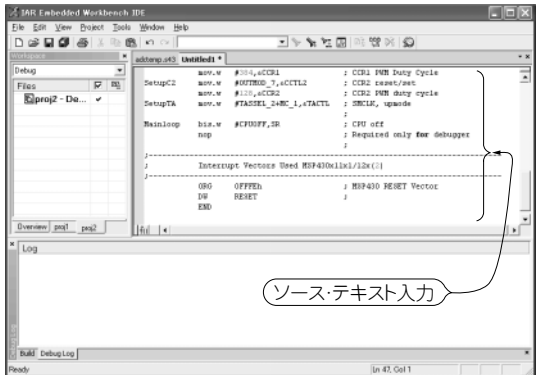

図3-26　PWM 波形出力ソース・プログラムのテキスト入力完了画面

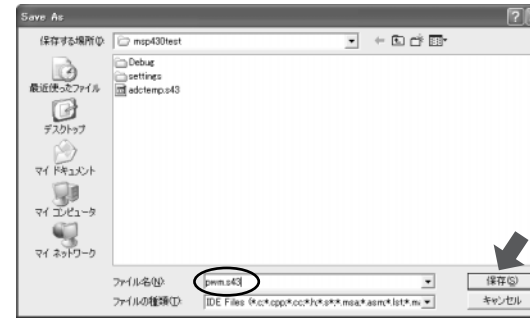

図3-27　ファイル名を pwm.s43 で保存

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

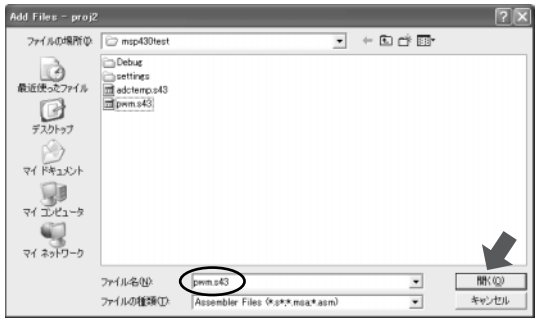

図3-28　ソース・ファイルをプロジェクトProj2へ登録

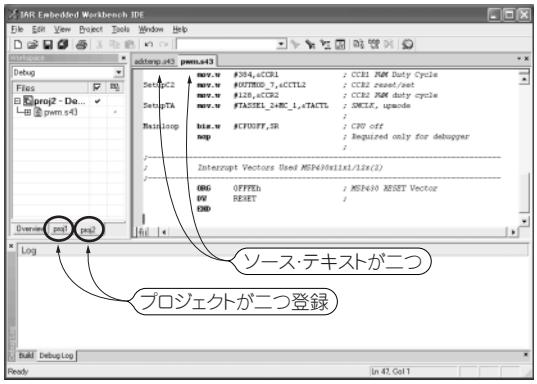

図3-29　プロジェクトに登録された画面

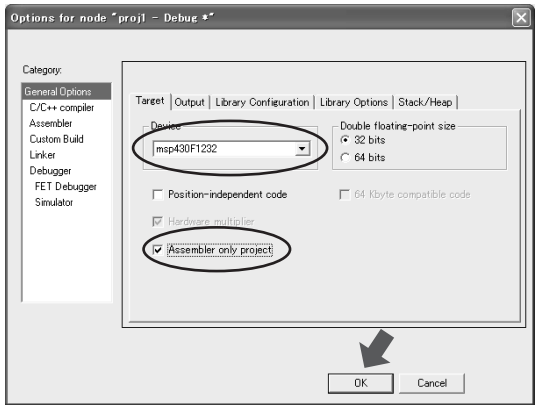

図3-30　「General Options」の設定

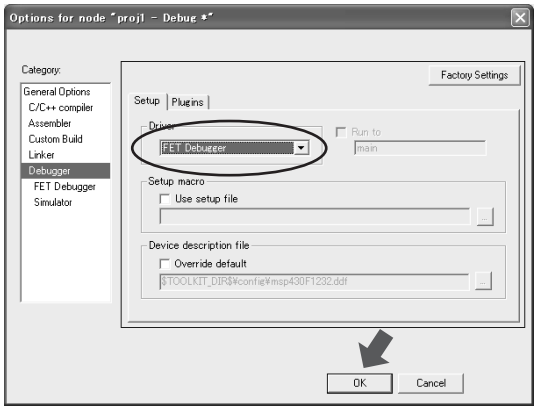

図3-31　「FET Debugger」を選択

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

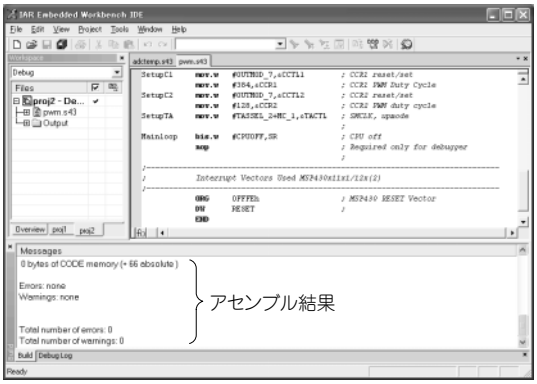

図3-32　アセンブル結果画面

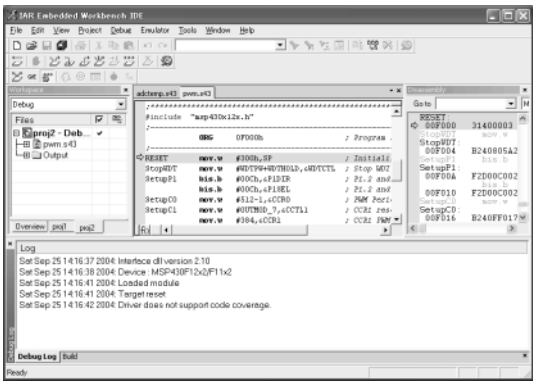

図3-33　転送完了画面

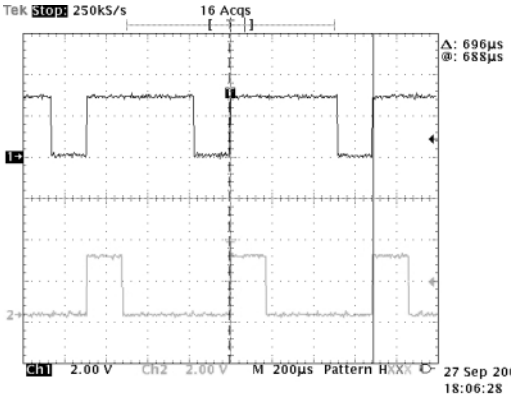

写真3-3　ポートP1.2，P1.3で観測した75％と25%のPWM出力波形（200 µs/div）

図3-34　終了時のプロジェクト保存画面

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．

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  "msp430x12x.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   #TASSEL_2+MC_1,&TACTL   ; 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

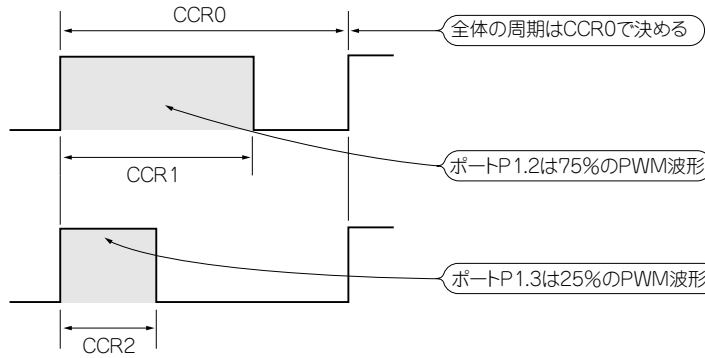

図3-35　各パルスの幅

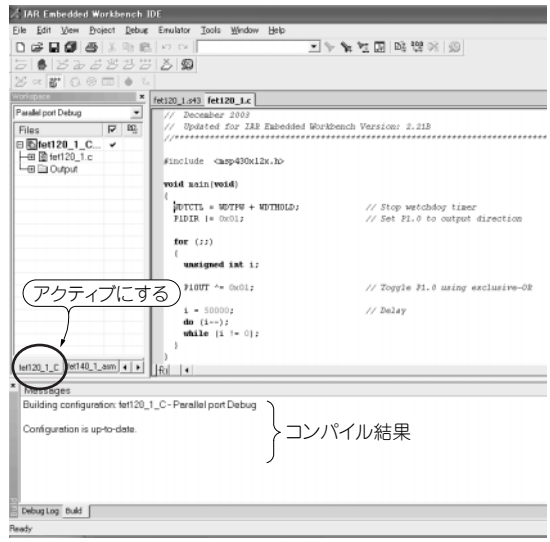

図3-36　C言語で作られたLED点滅サンプル・プログラムのコンパイル結果

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．