Appendix 2

カラーLEDの色を自由に調整する

Applilet を使って簡単開発

鈴木 康之

Applilet はマイコンの初期設定プログラムを自動生成するツールです．Applilet については下記 URL を参照してください．

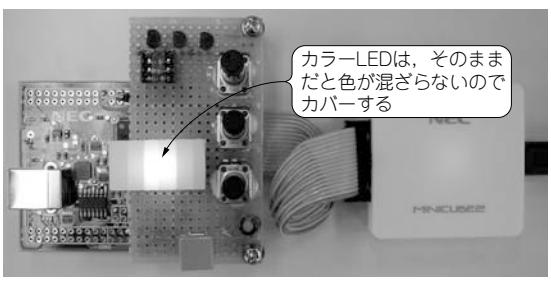
http://www.necel.com/micro/ja/development/asia/applilet/

Applilet の使い方については，第10章を参照してください．

このツールと MINICUBE2 を使い，簡単にマイコン・プログラムを開発できる手法を紹介します．

Applilet を使用して，カラーLEDの色を自由に調整するシステムを作成してみましょう（**図A**）．回路図を**図B**に，部品表を**表A**に示します．半固定抵抗の値によりPWMタイマのデューティ比を変化させ，赤，青，緑LEDの輝度を調整します．3色を調整すれば，パステル調の色を作ることも可能です．スイッチの押下によりV850基板上のLEDが点灯/消灯し，消灯時に輝度調整が有効になります．

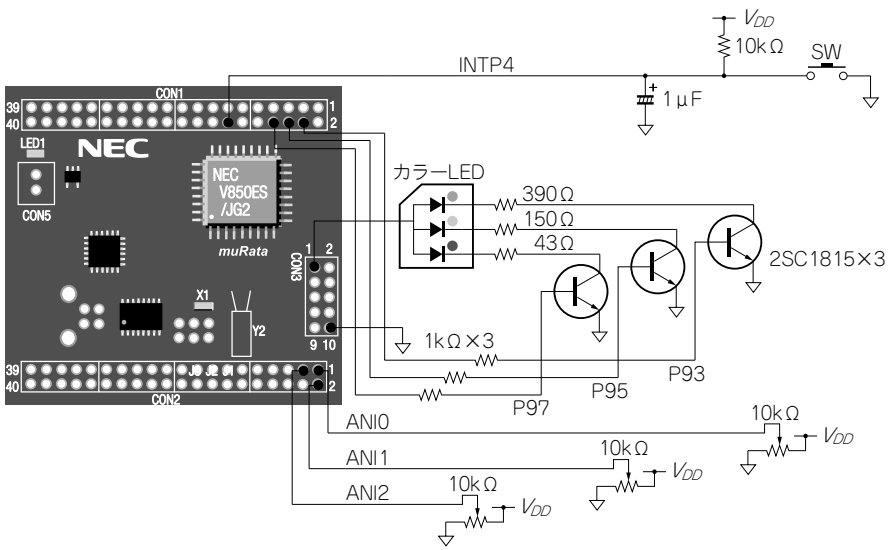
では，Applilet を起動してみましょう．はじめに，新規作成の画面でシリーズ名「V850ESJG2」とチップ名「μPD70F3716」を選択し，新規プロジェクトを作成します．そしてシステム（**図C**）や割り込み（**図D**），ポート（**図E**），A-Dコンバータ（**図F**），タイマP0（イ



図A　カラーLEDの色を自由に調整する装置

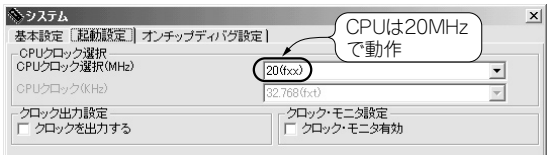
表A　部品表

No.	部品名	個数	備考
1	フル・カラーLED	1	
2	押しボタン・スイッチ	1	
3	電解コンデンサ（1μF）	1	
4	2SC1815	3	
5	半固定抵抗（10kΩ）	3	
6	抵抗（10kΩ）	1	
7	抵抗（1kΩ）	3	
8	抵抗（390Ω）	1	赤色LED用
9	抵抗（150Ω）	1	青色LED用
10	抵抗（43Ω）	1	緑色LED用



図B
カラーLED色調整用の回路図

（a）基本設定の画面



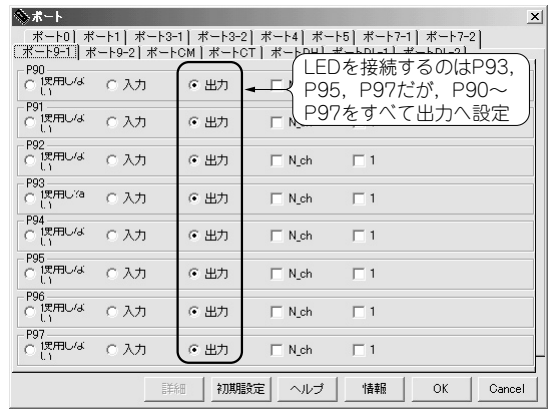
（b）起動設定の画面

（c）オンチップディバグ設定の画面

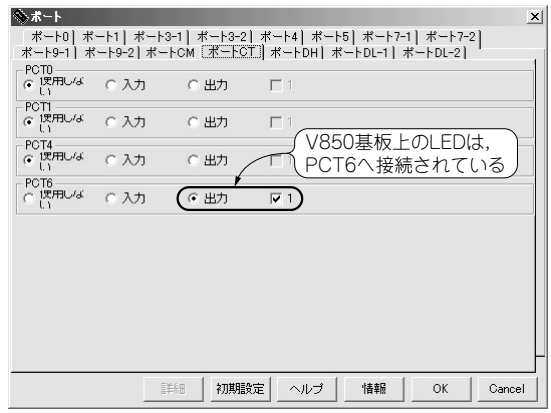
図 C　システムの設定画面

図 D　割り込みの設定画面

図 F　A-D コンバータの設定画面



（a）ポート9-1の画面



（b）ポートCTの画面

図 E　ポートの設定画面

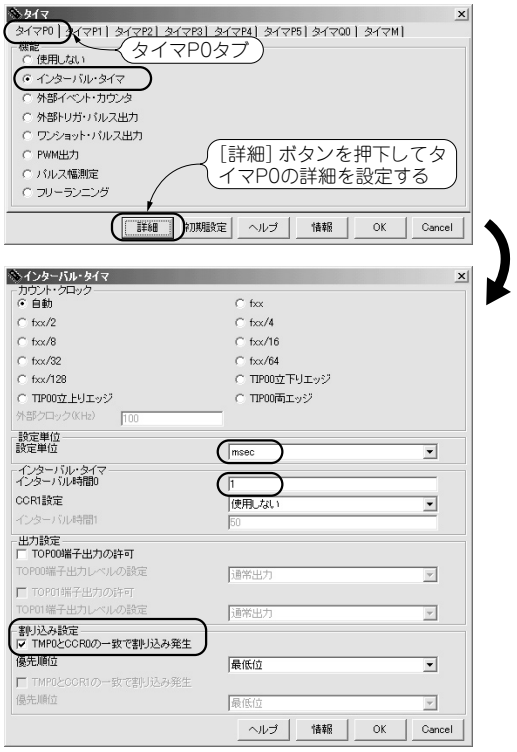


図 G　タイマ P0 の設定画面

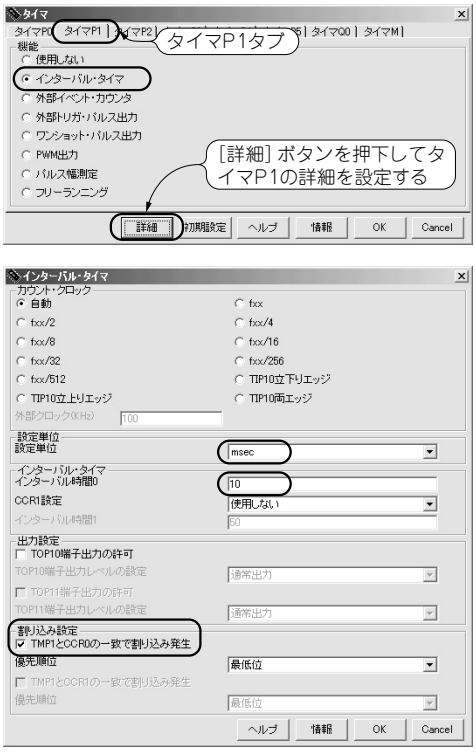
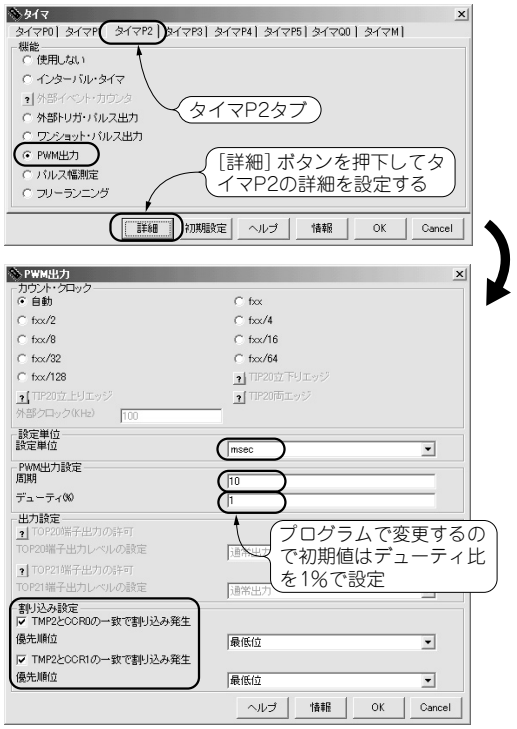


図 H　タイマ P1 の設定画面



※同様にタイマP3，タイマP4もPWM出力の設定を行う（設定単位10msec，デューティ 1%，割り込み設定も2個所にチェックする）.

図 I　タイマ P2 の設定画面

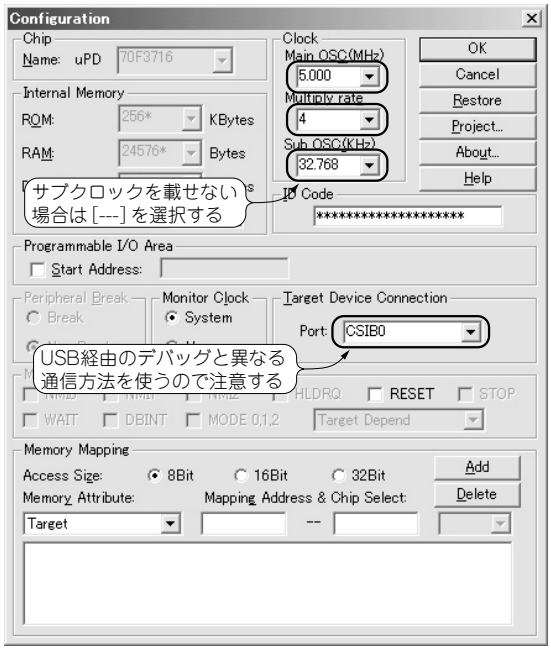


図 J　デバッガの起動画面

リスト A　main.c のソース（追加部分）

```
～省略～
/*
************************************************
** MacroDefine
************************************************
*/
UINT   g_tmp0counter;
UINT   g_tmp1counter;
USHORT g_adtmp[ 3 ];
USHORT g_advalue[ 3 ];
UCHAR  g_ledflag[ 3 ];
UCHAR  g_automode;
USHORT g_autoledcounter;

void init_value( void )
{
    int i;
    for ( i = 0; i < 3; i++ )
    {
        g_adtmp[ i ] = 0;
        g_advalue[ i ] = 0;
        g_ledflag[ i ] = 0;
     }
     g_automode = 0;
     g_tmp0counter = 0;
     g_tmp1counter = 0;
     g_autoledcounter = 0;
}

/*
**-----------------------------------------------
** Abstract: main function
** Parameters: None
** Returns: None
**-----------------------------------------------
*/
void main( void )
{
    TMP0_Start();
    TMP1_Start();
    TMP2_Start();
    TMP3_Start();
    TMP4_Start();
    while (1) {
        ;
    }
}
```

リスト B　ad_user.c のソース（追加部分）

```
～省略～
/*
************************************************
** MacroDefine
************************************************
*/
extern USHORT g_adtmp[ 3 ];
extern USHORT g_advalue[ 3 ];
extern UCHAR g_ledflag[ 3 ];

/*
**-----------------------------------------------
** Abstract: INTAD Interrupt service routine
** Parameters: None
** Returns: None
**-----------------------------------------------
*/
__interrupt void MD_INTAD( void )
{
    int i;
    AD_Read( &g_adtmp[ 0 ] );
    /* A-D値をまるめる．A-D値に変化があれば
                                LED点灯フラグをセットする．*/
    for ( i = 0; i < 3; i++ )
    {
        g_adtmp[ i ] =
                    ( g_adtmp[ i ] & 0xfff ) >> 3;
        if ( g_advalue[ i ] != g_adtmp[ i ] )
        {
            g_advalue[ i ] = g_adtmp[ i ];
            g_ledflag[ i ] = MD_ON;
        }
    }
}
```

リスト D　int_user.c のソース（追加部分）

```
～省略～
/*
************************************************
** MacroDefine
************************************************
*/
extern UCHAR g_automode;

/*
**-----------------------------------------------
** Abstract: This function is INTP4 Interrupt service
routine.
** Parameters: None
** Returns: None
**-----------------------------------------------
*/
__interrupt void MD_INTP4( void )
{
    /* SW押下時に発生する割り込み処理*/
    g_automode ^= 1;
    if ( g_automode == 0 )
    {
        PCT.6 = 1;          /* V850基板のLED消灯*/
    }
    else
    {
        PCT.6 = 0;          /* V850基板のLED点灯*/
    }
}
```

リスト C　timer_user.c のソース（追加部分）

```
/***************************************************
**  Include files
***************************************************/

#include "macrodriver.h"
#include "timer.h"
#include "ad.h"
#pragma interrupt INTTP0CC0 MD_INTTP0CC0
#pragma interrupt INTTP1CC0 MD_INTTP1CC0
#pragma interrupt INTTP2CC0 MD_INTTP2CC0
#pragma interrupt INTTP2CC1 MD_INTTP2CC1
#pragma interrupt INTTP3CC0 MD_INTTP3CC0
#pragma interrupt INTTP3CC1 MD_INTTP3CC1
#pragma interrupt INTTP4CC0 MD_INTTP4CC0
#pragma interrupt INTTP4CC1 MD_INTTP4CC1

/***************************************************
**  Constants
***************************************************/

extern UINT g_tmp0counter;
extern UINT g_tmp1counter;
extern USHORT g_advalue[ 3 ];
extern UCHAR  g_ledflag[ 3 ];
extern UCHAR  g_automode;
extern USHORT g_autoledcounter;

USHORT g_dutydata[ 3 ][ 2 ] =
{
    { TM_TMP2_PWMCYCLE, 0 }
  , { TM_TMP3_PWMCYCLE, 0 }
  , { TM_TMP4_PWMCYCLE, 0 }
};

UCHAR g_autodutydata[] =
{   /* R, G, B データのデータを255個 */
  7,  7,124, 10, 10,121, 13, 13,118, 16, 16,115,
 19, 19,112, 22, 22,109, 25, 25,106, 28, 28,103,
 31, 31,100, 34, 34, 97, 37, 37, 94, 40, 40, 91,
 43, 43, 88, 46, 46, 85, 49, 49, 82, 52, 52, 79,
 55, 55, 76, 58, 58, 73, 61, 61, 70, 64, 64, 67,
 67, 67, 64, 70, 70, 61, 73, 73, 58, 76, 76, 55,
 79, 79, 52, 82, 82, 49, 85, 85, 46, 88, 88, 43,
 91, 91, 40, 94, 94, 37, 97, 97, 34,100,100, 31,
103,103, 28,106,106, 25,109,109, 22,112,112, 19,
115,115, 16,118,118, 13,121,121, 10,124,124,  7,

124,124,  7,121,121, 10,118,118, 13,115,115, 16,
112,112, 19,109,109, 22,106,106, 25,103,103, 28,
100,100, 31, 97, 97, 34  94, 94, 37, 91, 91, 40,
 88, 88, 43, 85, 85, 46, 82, 82, 49, 72, 79, 52,
 76, 76, 55, 73, 73, 58, 70, 70, 61, 67, 67, 64,
 64, 64, 67, 61, 61, 70, 58, 58, 73, 56, 55, 76,
 52, 52, 79, 49, 49, 82, 46, 46, 85, 43, 43, 88,
 40, 40, 91, 37, 37, 94, 34, 34, 97, 30, 31,100,
 28, 28,103, 25, 25,106, 22, 22,109, 19, 19,112,
 16, 16,115, 13, 13,118, 10, 10,121,  7,  7,124,

124,  7,124,121, 10,121,118, 13,118,115, 16,115,
112, 19,112,109, 22,109,106, 25,106,103, 28,103,
100, 31,100, 97, 34, 97, 94, 37, 94, 91, 40, 91,
 88, 43, 88, 85, 46, 85, 82, 49, 82, 72, 52, 79,
 76, 55, 76, 73, 58, 73, 70, 61, 70, 67, 64, 67,
 64, 67, 64, 61, 70, 61, 58, 73, 58, 56, 76, 55,
 52, 79, 52, 49, 82, 49, 46, 85, 46, 43, 88, 43,
 40, 91, 40, 37, 94, 37, 34, 97, 34, 30,100, 31,
 28,103, 28, 25,106, 25, 22,109, 22, 19,112, 19,
 16,115, 16, 13,118, 13, 10,121, 10,  7,124,  7,

  7,124,  7, 10,121, 10, 13,118, 13, 16,115, 16,
 19,112, 19, 22,109, 22, 25,106, 25, 28,103, 28,
 31,100, 31, 34, 97, 34, 37, 94, 37, 40, 91, 40,
 43, 88, 43, 46, 85, 46, 49, 82, 49, 52, 79, 52,
 55, 76, 55, 58, 73, 58, 61, 70, 61, 64, 67, 64,
 67, 64, 67, 70, 61, 70, 73, 58, 73, 76, 55, 76,
 79, 52, 79, 82, 49, 82, 85, 46, 85, 88, 43, 88,
 91, 40, 91, 94, 37, 94, 97, 34, 97,100, 31,100,
103, 28,103,106, 25,106,109, 22,109,112, 19,112,
115, 16,115,118, 13,118,121, 10,121,124,  7,124,
  7,124,124, 10,121,121, 13,118,118, 16,115,115,
 19,112,112, 22,109,109, 25,106,106, 28,103,103,
 31,100,100, 34, 97, 97, 37, 94, 94, 40, 91, 91,
 43, 88, 88, 46, 85, 85, 49, 82, 82, 52, 72, 79,
 55, 76, 76, 58, 73, 73, 61, 70, 70, 64, 67, 67,
 67, 64, 64, 70, 61, 61, 73, 58, 58, 76, 56, 55,
 79, 52, 52, 82, 49, 49, 85, 46, 46, 88, 43, 43,
 91, 40, 40, 94, 37, 37, 97, 34, 34,100, 30, 31,
103, 28, 28,106, 25, 25,109, 22, 22,112, 19, 19,
115, 16, 16,118, 13, 13,121, 10, 10,124,  7,  7,

124,  7,  7,121, 10, 10,118, 13, 13,115, 16, 16,
112, 19, 19,109, 22, 22,106, 25, 25,103, 28, 28,
100, 31, 31, 97, 34, 34, 94, 37, 37, 91, 40, 40,
 88, 43, 43, 85, 46, 46, 82, 49, 49, 79, 52, 52,
 76, 55, 55, 73, 58, 58, 70, 61, 61, 67, 64, 64,
 64, 67, 67, 61, 70, 70, 58, 73, 73, 55, 76, 76,
 52, 79, 79, 49, 82, 82, 46, 85, 85, 43, 88, 88,
 40, 91, 91, 37, 94, 94, 34, 97, 97, 31,100,100,
 28,103,103, 25,106,106, 22,109,109, 19,112,112,
 16,115,115, 13,118,118, 10,121,121,  7,124,124,

  7,127,127,  7,127,127,  7,127,127,  7,127,127,
  7,127,127,  7,127,127,  7,127,127,  7,127,127
};

__interrupt void MD_INTTP0CC0( void )
{
    /* 1msec ごとの割り込み処理 */
    g_tmp0counter++;
}

__interrupt void MD_INTTP1CC0( void )
{
    USHORT ustmp;
    int i;
    g_tmp1counter++; /* 10msec ごとの割り込み処理 */

    if ( g_automode == 1 ) /* 自動点灯モード・チェック */
    {
        if ( g_tmp1counter > 2 )
        {   /* 20msec ごとに色を変える */
            g_tmp1counter = 0;
            ustmp = g_autoledcounter & 0xff;
            if ( ustmp & 0x80 )
            {
                ustmp &= 0x7f;
                ustmp = 0x7f - ustmp;
            }

            /* duty データ・リセット */
            for ( i = 0; i < 3; i++ )
            {
                g_dutydata[ i ][ 1 ] = 0;
            }

            /* LED 点滅パターン設定 */
            if ( ( g_autoledcounter & 0x700 ) == 0 )
            {
                ustmp = g_autoledcounter & 0xff;
                g_dutydata[0][1]=g_autodutydata
                                    [ustmp*3+0]*0x200;
                g_dutydata[1][1]=g_autodutydata
                                    [ustmp*3+1]*0x200;
                g_dutydata[2][1]=g_autodutydata
                                    [ustmp*3+2]*0x200;
            }
            else
            {
                if ( g_autoledcounter & 0x100 )
                                        /* 赤 LED セット */
                {
                    g_dutydata[ 0 ][ 1 ]
                                        = ustmp * 0x200;
                }

                if ( g_autoledcounter & 0x200 )
                                        /* 緑 LED セット */
                {
                    g_dutydata[ 1 ][ 1 ]
```

リスト C　timer_user.c のソース（追加部分）

```
                                        = ustmp * 0x200;
                }

                if ( g_autoledcounter & 0x400 )
                                        /* 青 LED セット */
                {
                    g_dutydata[ 2 ][ 1 ]
                                        = ustmp * 0x200;
                }
            }

            /* LED 各色デューティ比変更 */
            if ( g_dutydata[ 0 ][ 1 ] != 0 )
            {
                TMP2_ChangeTimerCondition
                                (&g_dutydata[0][0],2);
            }
            if ( g_dutydata[ 1 ][ 1 ] != 0 )
            {
                TMP3_ChangeTimerCondition
                                (&g_dutydata[1][0],2);
            }
            if ( g_dutydata[ 2 ][ 1 ] != 0 )
            {
                TMP4_ChangeTimerCondition
                                (&g_dutydata[2][0],2);
            }

            /* LED 点灯フラグをセット */
            for ( i = 0; i < 3; i++ )
            {
                g_ledflag[ i ] = 1;
            }
            g_autoledcounter++;
        }
    }
    else
    {
        AD_Start(); /* A-D 変換処理開始 */
    }
}

__interrupt void MD_INTTP2CC0( void )
{
    /* 赤 LED の値が変わったらデューティ比を変更する */
    if ( ( g_ledflag[ 0 ] != 0 ) &&
                                ( g_automode == 0 ) )
    {
        g_ledflag[ 0 ] = 0;
        g_dutydata[ 0 ][ 1 ]
                = TM_TMP2_PWMWIDTH * g_advalue[ 0 ];
        TMP2_ChangeTimerCondition
                                ( &g_dutydata[ 0 ][ 0 ], 2 );
    }
    P9L.7 = 1;
}

__interrupt void MD_INTTP2CC1( void )
{
    P9L.7 = 0;
}

__interrupt void MD_INTTP3CC0( void )
{
    /* 緑 LED の値が変わったらデューティ比を変更する */
    if ( ( g_ledflag[ 1 ] != 0 ) &&
                                ( g_automode == 0 ) )
    {
        g_ledflag[ 1 ] = 0;
        g_dutydata[ 1 ][ 1 ]
                = TM_TMP4_PWMWIDTH * g_advalue[ 1 ];
        TMP3_ChangeTimerCondition
                                ( &g_dutydata[ 1 ][ 0 ], 2 );
    }
    P9L.5 = 1;
}

__interrupt void MD_INTTP3CC1( void )
{
    P9L.5 = 0;
}

__interrupt void MD_INTTP4CC0( void )
{
    /* 青 LED の値が変わったらデューティ比を変更する */
    if ( ( g_ledflag[ 2 ] != 0 ) &&
                                ( g_automode == 0 ) )
    {
        g_ledflag[ 2 ] = 0;
        g_dutydata[ 2 ][ 1 ]
                = TM_TMP4_PWMWIDTH * g_advalue[ 2 ];
        TMP4_ChangeTimerCondition
                                ( &g_dutydata[ 2 ][ 0 ], 2 );
    }
    P9L.3 = 1;
}

__interrupt void MD_INTTP4CC1( void )
{
    P9L.3 = 0;
}
```

ンターバル・タイマ）（**図 G**），タイマ P1（インターバル・タイマ）（**図 H**），タイマ P2（PWM 出力，**図 I**）のそれぞれを設定します．設定が完了したら，コードを自動生成します．

生成されたコードを PM+（統合開発環境）を使用して編集します．**リスト A**～**リスト D** に Applilet が生成したソースから変更した点を示します．ソース編集後に PM+ でビルドしてエラーがないことを確認します．

デバッグには統合デバッガ ID850QB を使用します．PM+ でデバッガを起動させると**図 J** が表示されるのでそれぞれ変更し，［OK］ボタンを押下してください．プログラムがダウンロードされたら［GO］ボタンで実行します．

すずき・やすゆき
NEC エレクトロニクス（株）