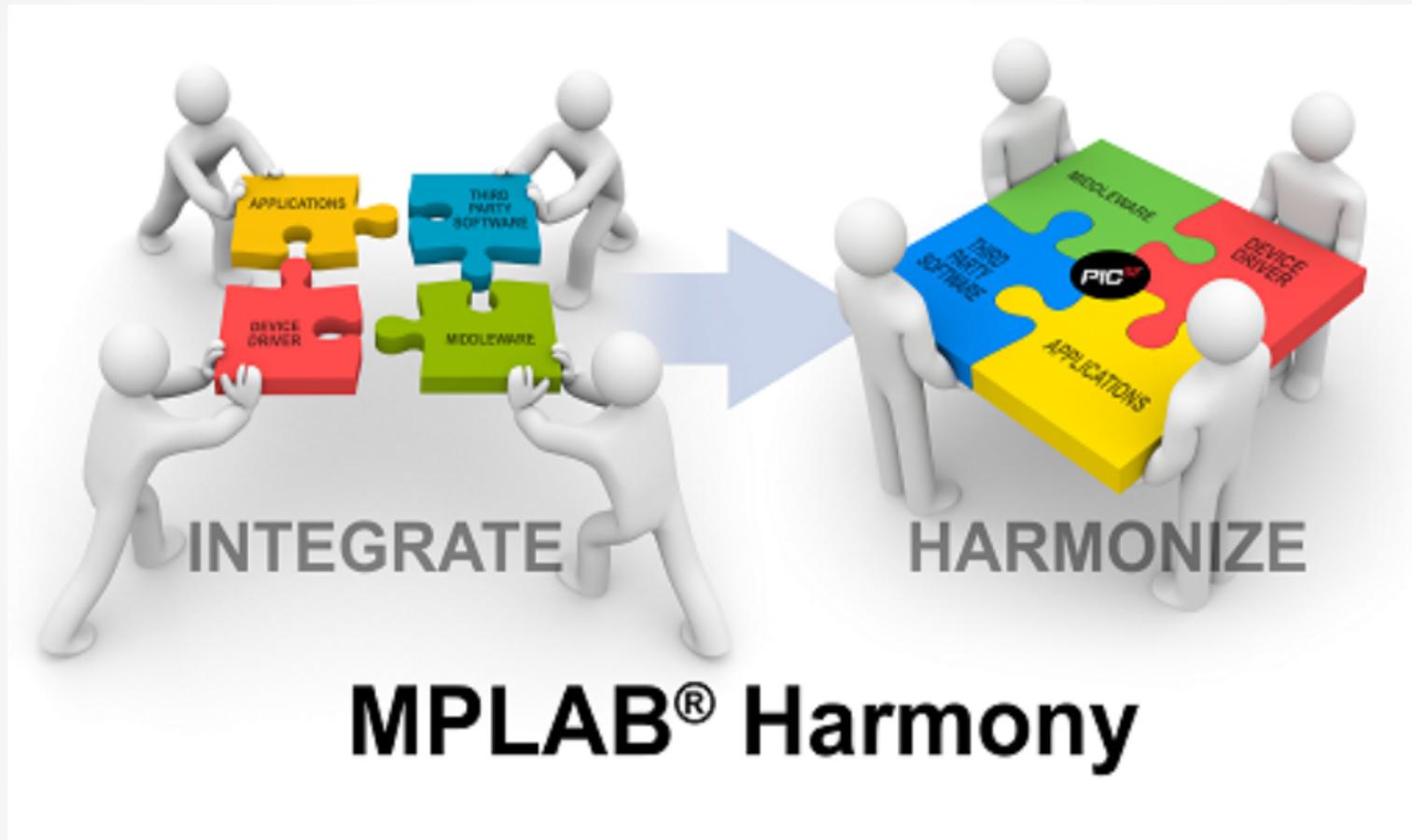


HarmonyでLチカ



果たして、HarmonyはPIC32MXの敷居を下げているのか？

前回(TQFPの半田付け)の課題

Oscillogi2のプログラムをHarmonyベースで書き換える。

==> 作業は順調に進んでいる。

Oscillogi2のコンソール・プログラムを書き換える。
(オシロ7CH、ロジアナ12CHへの対応)

==> 作業は難航（前進は）している。

Oscillogi2のHarmonize(?)の話より
Harmony導入時に行ったLチカの話の方が良いかも・・・

これがやる気を削いでしまう



MPLAB Harmony Help

MPLAB Harmony Integrated Software Framework v1.05

Harmoyのドキュメントはこれ(8400ページもある)しか無い！(？)

どこから手をつけるか

Where to Begin

The help documentation provides a comprehensive source of information on how to use and understand MPLAB Harmony. However, you don't need to read the entire document before you start working with MPLAB Harmony. Where you should begin depends on your current objectives and level of familiarity with MPLAB Harmony.

Start here...	If you...
Release Contents	...want to know what is included in this release.
Release Notes	...want to know what is new in this release and to learn of any known issues.
Understanding MPLAB Harmony	...are new to MPLAB Harmony and need to understand the basic concepts.
<u>Prerequisites</u>	...want to make sure that you have everything you need to begin working with MPLAB Harmony.
Application and Library Development, Distribution, and Integration	...want to know how to develop MPLAB Harmony-compatible applications and libraries and how to best distribute and integrate them into an existing installation.
Tutorial	...are ready to start creating your own MPLAB Harmony applications.
Porting to MPLAB Harmony	...are a MLA user and you need to port your application to MPLAB Harmony.

<u>Applications Help</u>	...want to build and use the demonstration and example applications included in the installation.
<u>Framework Help</u>	...want to look up details on how to use the MPLAB Harmony framework libraries.
Third-Party Products Help	...need help with one of the third-party products included in this installation.
Board Support Package Help	...want to look up details on how to use MPLAB Harmony Board Support Packages.
<u>Utilities Help</u>	...need help using the MPLAB Harmony Configurator (MHC) or one of the other utilities included in this installation.
MHC Developer's Guide	...want information on how to create your own MPLAB Harmony library and integrated help.
MPLAB Harmony Compatibility Guide	...want to ensure software libraries you create are compatible with MPLAB Harmony.
Testing Help	...want to know how to test your own MPLAB Harmony libraries.
Support	...need additional assistance.
Tips and Tricks	...want to learn more efficient and effective ways to use MPLAB Harmony.
Glossary	...need an explanation of the terms used in MPLAB Harmony.

The majority of the help content is located in the [Framework Help](#) section, which provides detailed reference information on the libraries that make up the MPLAB Harmony framework. Each library's help section also provides introductory information on how to build, configure, and use the library, along with the detailed interface reference. However, you do not need to read the Framework Help section from end-to-end to use MPLAB Harmony.

Tutorialは無視して良い。Framework Helpが8000ページある。

Harmonyを使う前の準備

MPLAB X IDE (最新版は3.10)

MPLAC XC32C/C++Compiler (最新版は1.40)

MPLAB Harmony (最新版は1.06)

MPLAB Harmony Configurator (最新版は1.06)

ただし、MPLAB Harmony ConfiguratorはMPLABXのPlugin

Helpの斜め読み

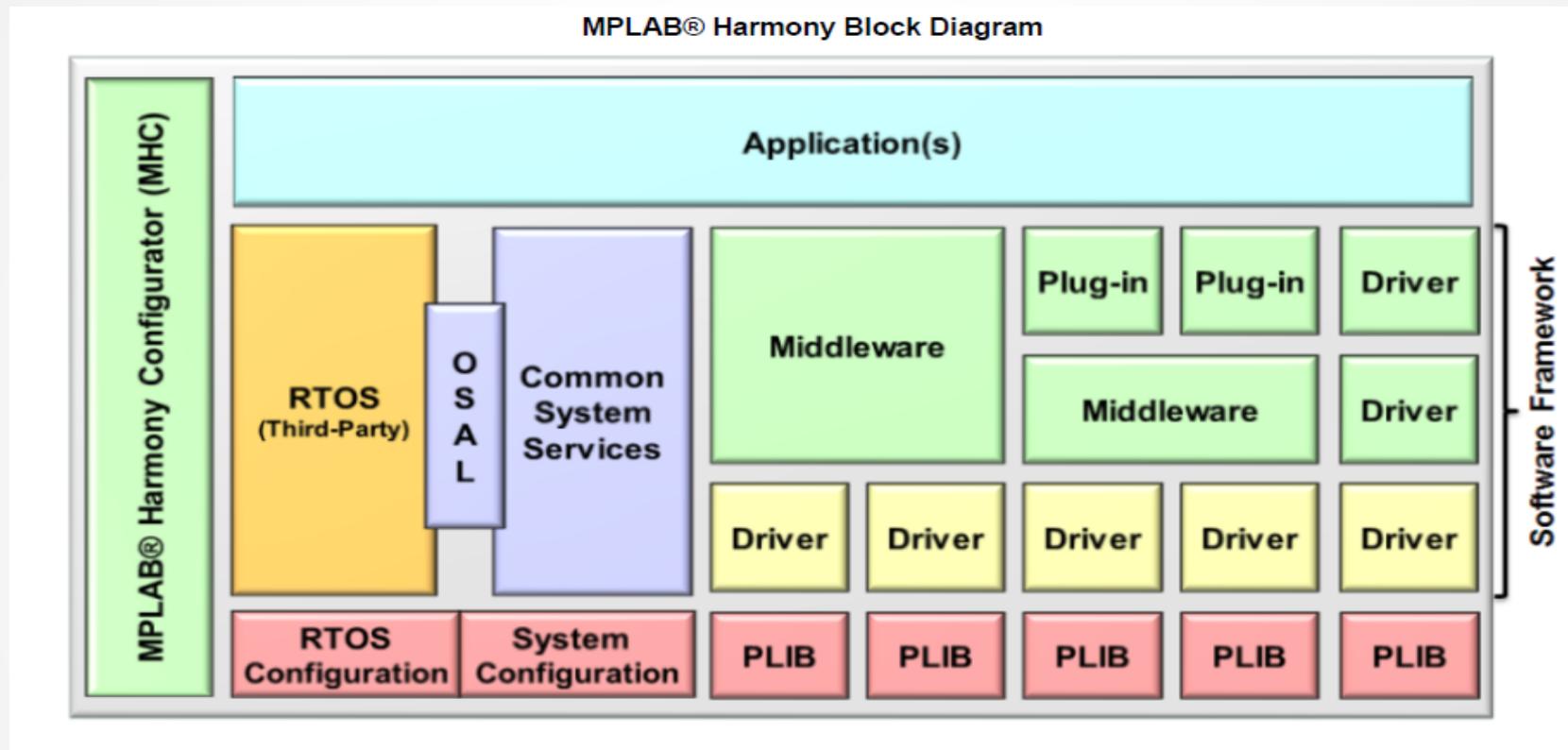
Applications Helpはサンプル・アプリケーションの説明

Framework Helpは各種ライブラリ説明

Utility HelpはMHCその他のUtilityの説明
MHC User's Guideに目を通しておけば
これから起きることの予習になる

v1_0x/apps以下のサンプル・プロジェクト
(特にexample/peripheral以下のサンプル) は
ライブラリを利用する上で大変参考になる

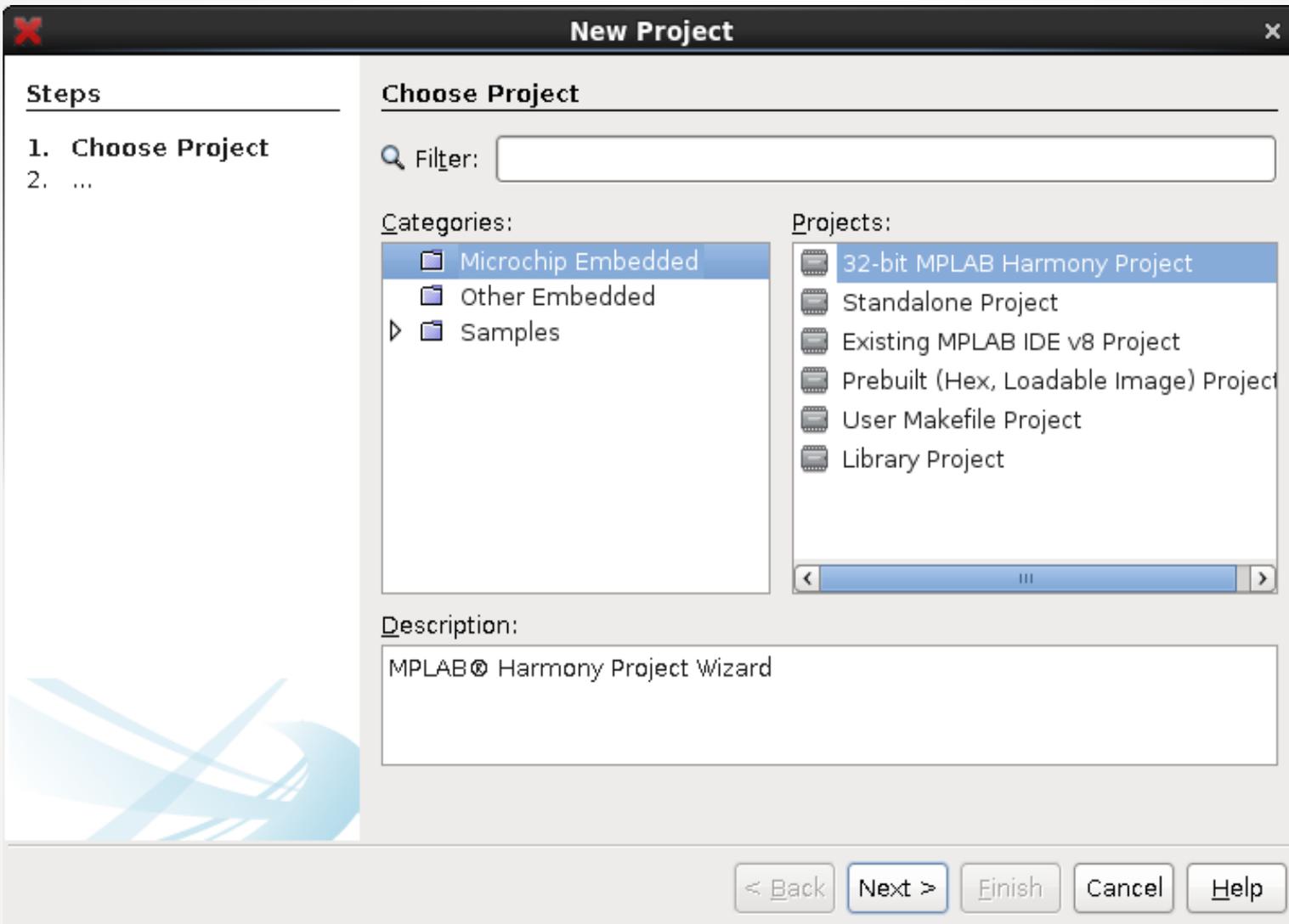
Harmonyの概要もざっと目を通しておく



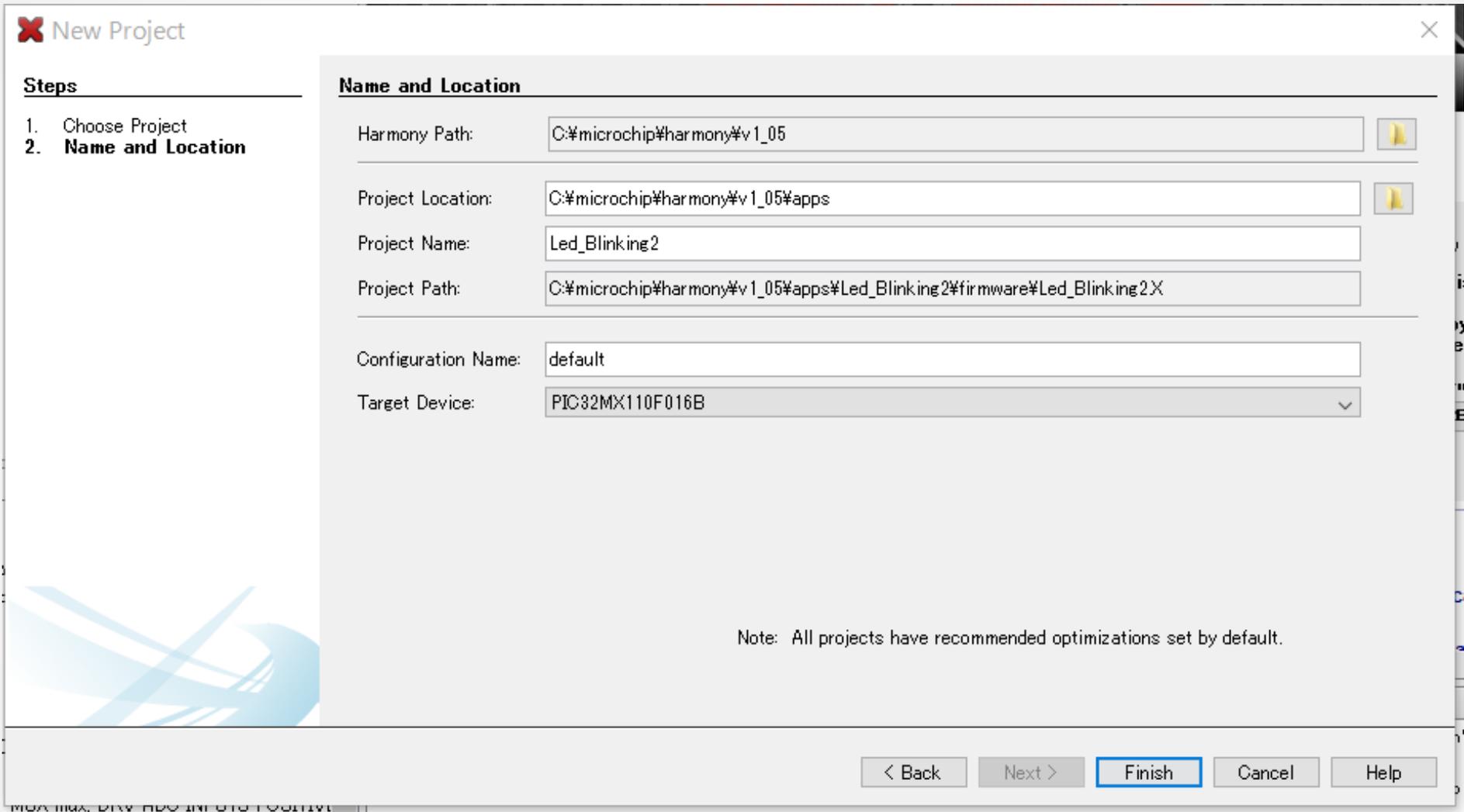
デバイス ドライバーは単純かつ高度に抽象化された
ペリフェラル インターフェース・・・

ペリフェラル ライブラリは低レベルのペリフェラル インター
フェースで、レジスタの詳細を隠すことによって、複数のコント
ローラ ファミリのサポートを容易にする・・・

新規プロジェクトを作る



Project名を決める



New Project

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Harmony Path: C:\microchip\harmony\v1_05

Project Location: C:\microchip\harmony\v1_05\apps

Project Name: Led_Blinking2

Project Path: C:\microchip\harmony\v1_05\apps\Led_Blinking2\firmware\Led_Blinking2

Configuration Name: default

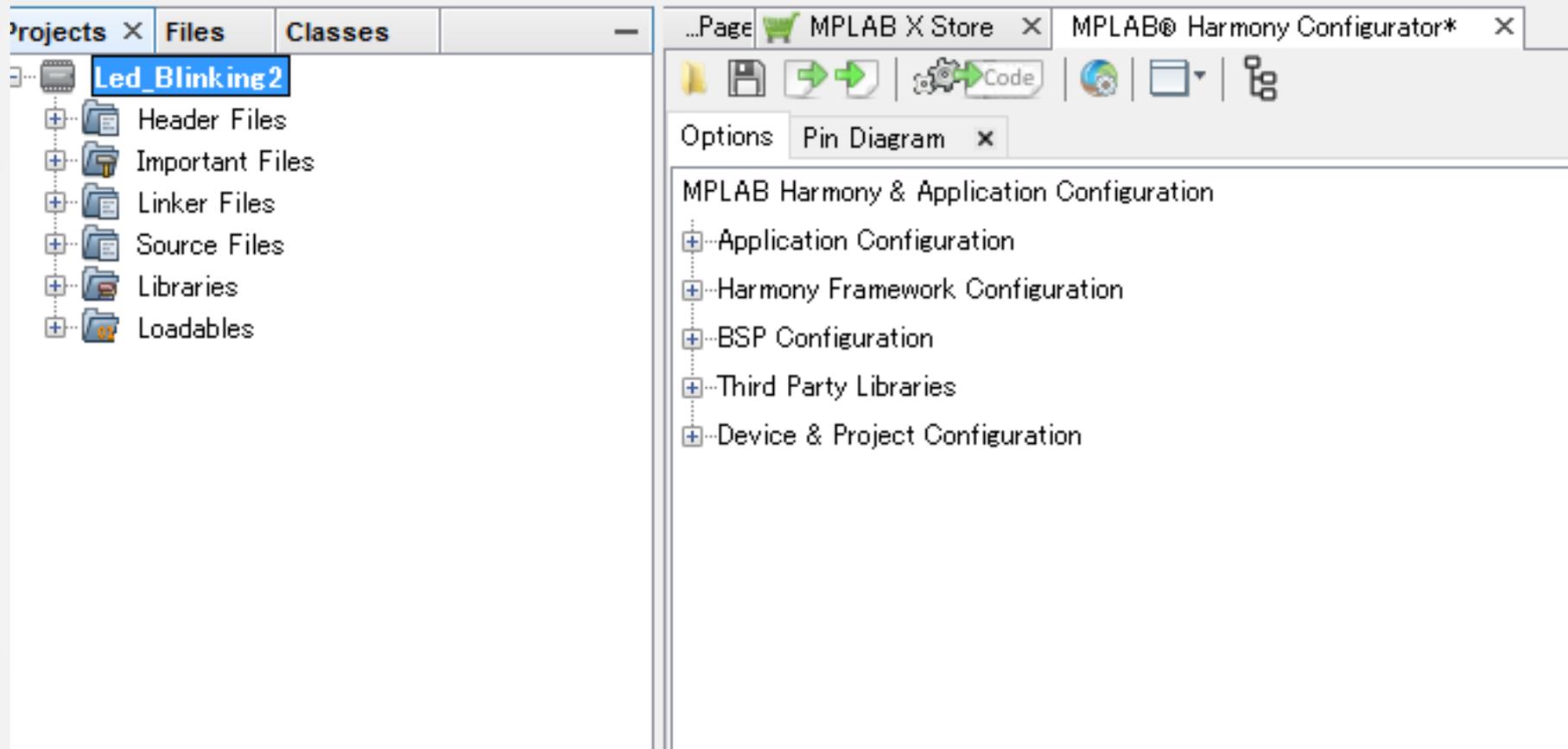
Target Device: PIC32MX110F016B

Note: All projects have recommended optimizations set by default.

< Back Next > **Finish** Cancel Help

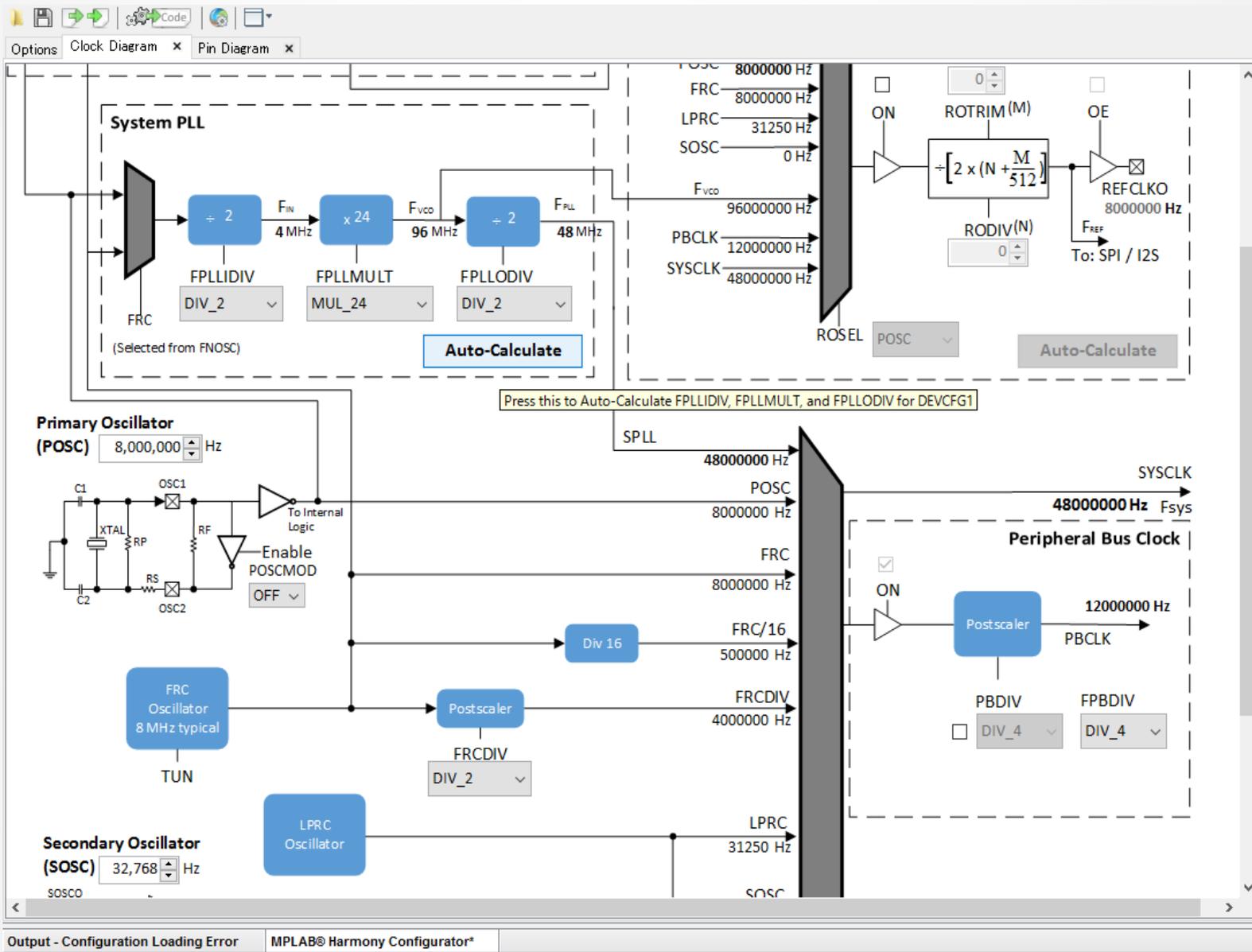
Target DeviceはProjectのPropertiesで設定しても良い

最初の状態

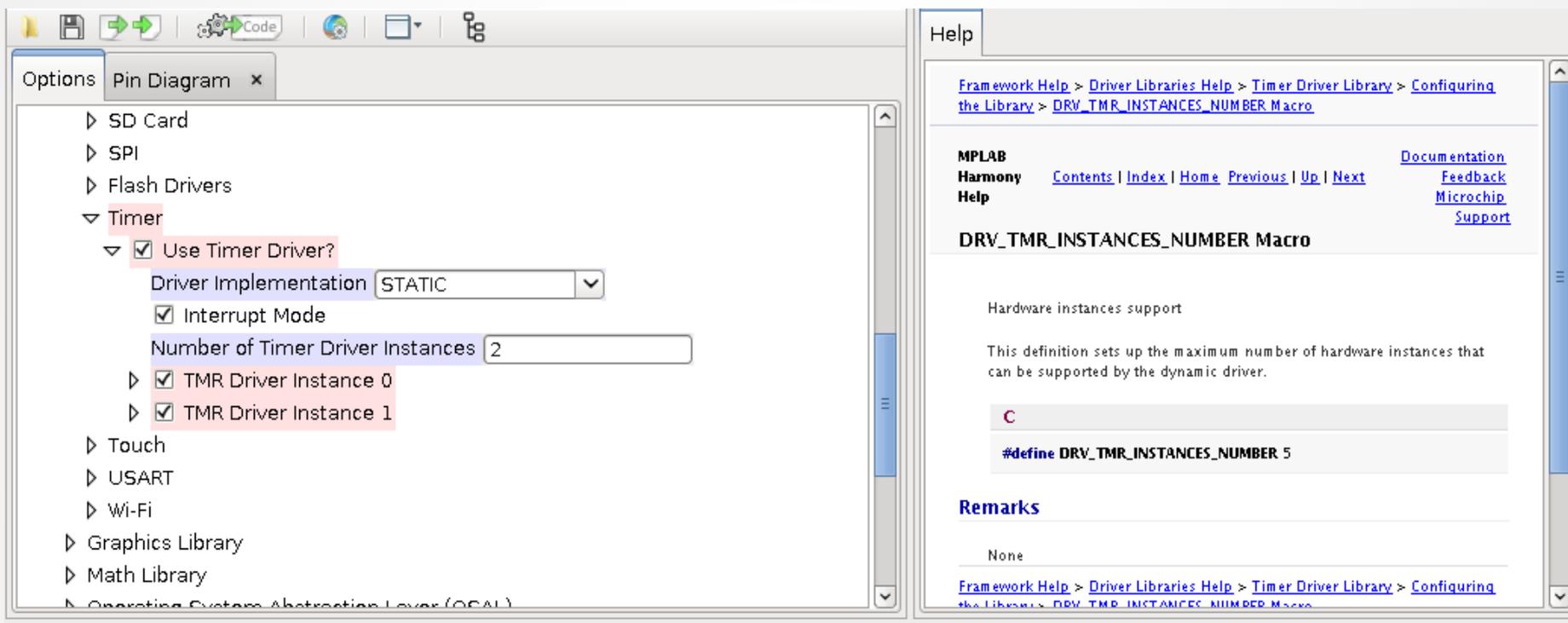


Projectフォルダは空っぽ

Clock Configurator



ProjectにTimer Driverを二つ組み込む

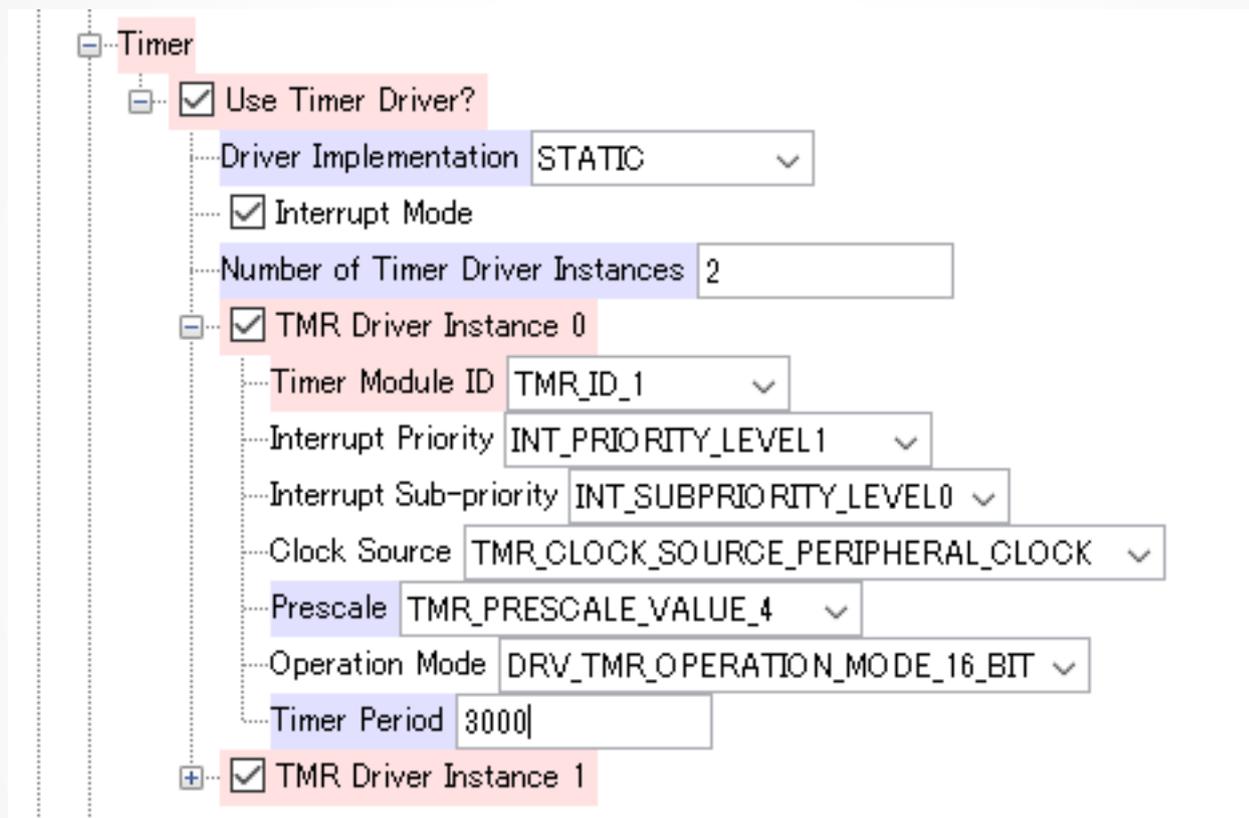


Harmony Framework Configuration

- > Drivers
- > Timer

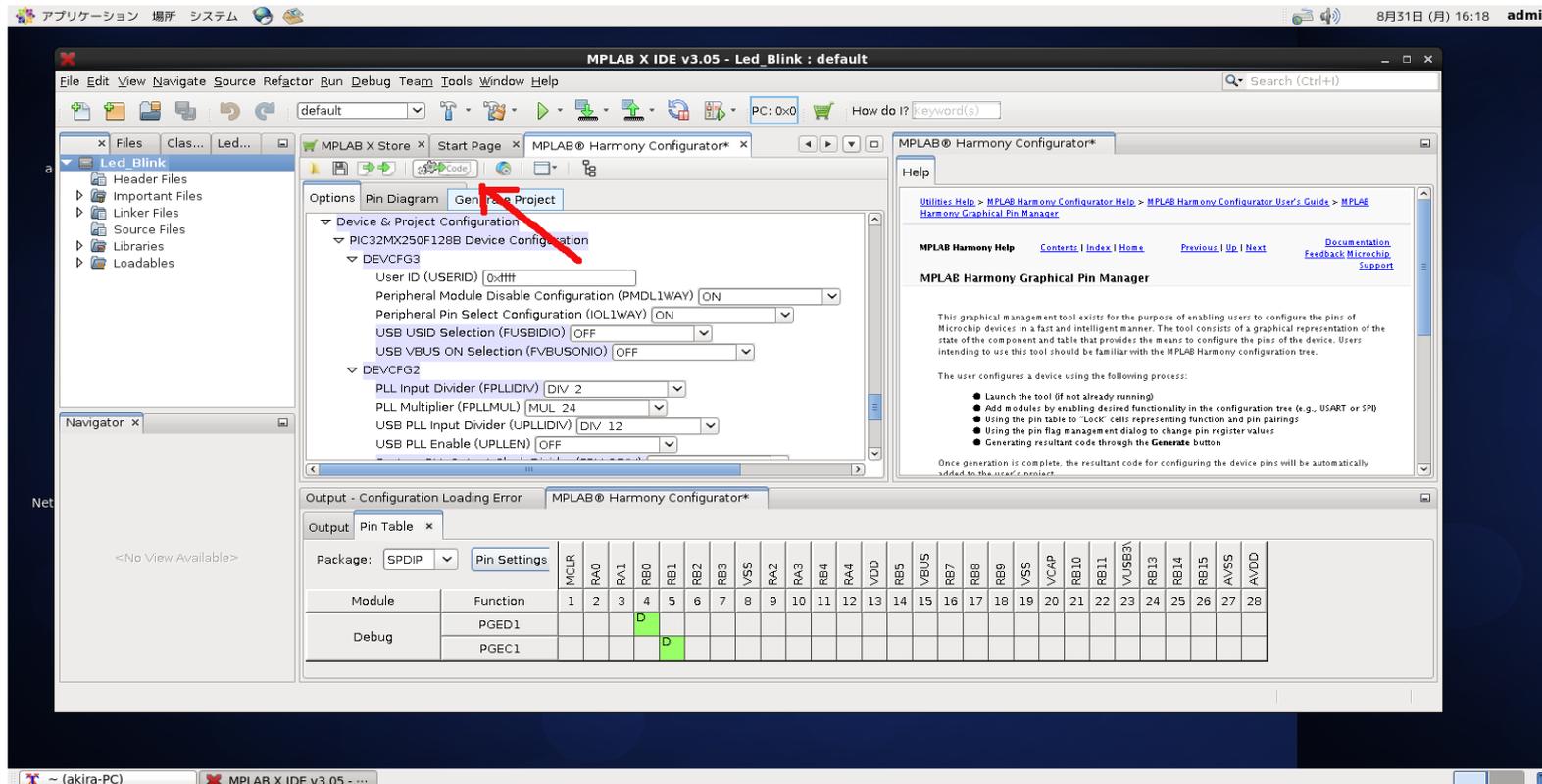
Use Timer Driver?にチェックを入れ、Interrupt Modeにチェックを入れ、Driver Instanceを2にした

Timer Driver Instance 0の設定



Timer Module IDに1 (Timer 1)を設定し、PrescaleにVALUE_4
Timer Periodに3000を設定した。ペリフェラル・クロックが12MHz
なら1msインターバルのタイマー割り込みになる

Generate Project



Harmony Framework Configuration

- > System Services
- > Clock
- > Ports

Device & Project Configurationなどを設定し、最後にGenerate Projectをクリックする

自動生成されるmain関数

```
67 // *****
68 // *****
69
70 int main ( void )
71 {
72     /* Initialize all MPLAB Harmony modules, including application(s). */
73     SYS_Initialize ( NULL );
74
75
76     while ( true )
77     {
78         /* Maintain state machines of all polled MPLAB Harmony modules. */
79         SYS_Tasks ( );
80
81     }
82
83     /* Execution should not come here during normal operation */
84
85     return ( EXIT_FAILURE );
86 }
87
88
89 /* *****
90 End of File
91 */
92
93
```

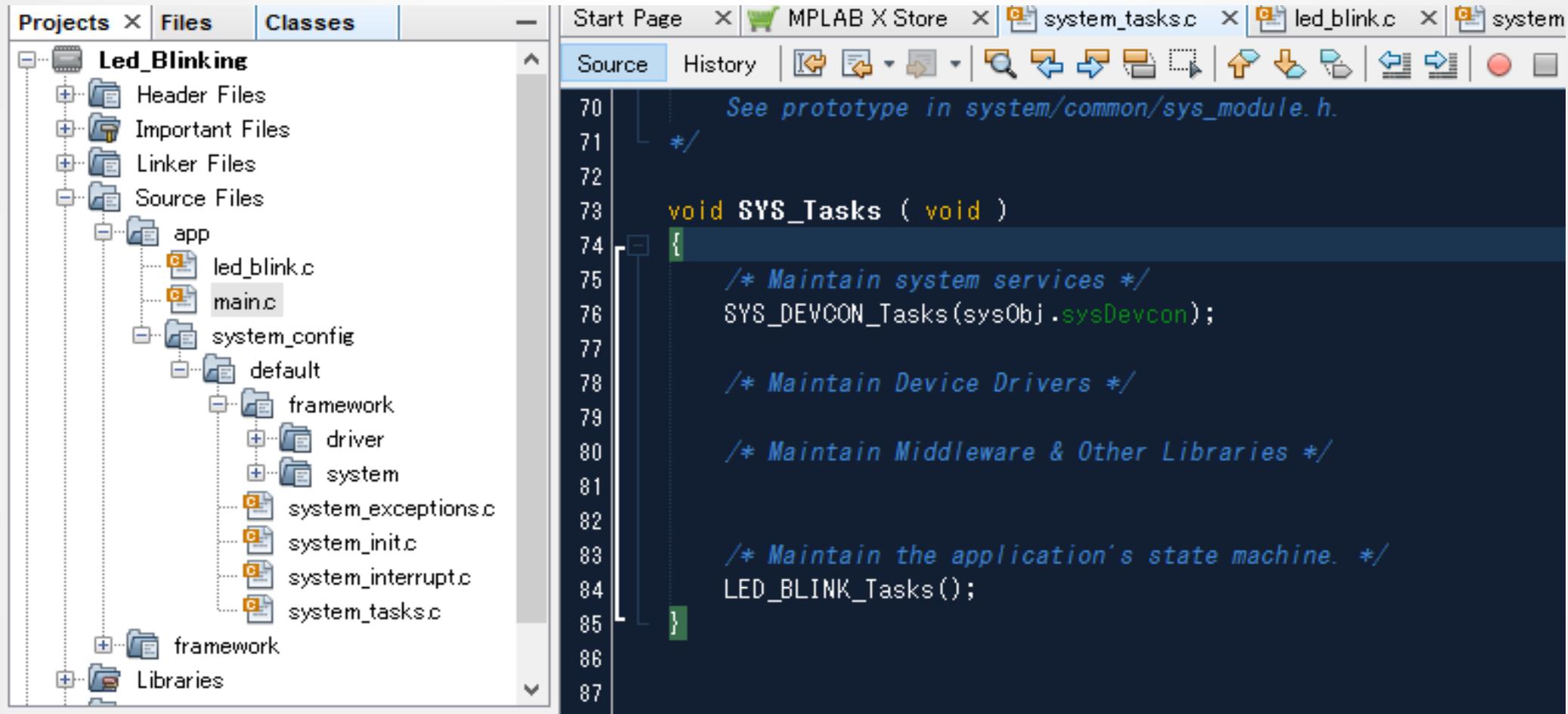
呼び出しているのは、SYS_Initialize()とSYS_Tasks()だけ

SYS_Initialize関数

```
174
175 void SYS_Initialize ( void* data )
176 {
177     /* Core Processor Initialization */
178     SYS_CLK_Initialize( NULL );
179     sysObj.sysDevcon = SYS_DEVCON_Initialize(SYS_DEVCON_INDEX_0, (SYS_MODULE_INIT*)&
180     SYS_DEVCON_PerformanceConfig(SYS_CLK_SystemFrequencyGet());
181     SYS_PORTS_Initialize();
182
183     /* Initialize Drivers */
184     /*Initialize TMRO */
185     DRV_TMRO_Initialize();
186     /*Initialize TMR1 */
187     DRV_TMR1_Initialize();
188
189
190     /* Initialize System Services */
191     SYS_INT_Initialize();
192
193     /* Initialize Middleware */
194     /* Enable Global Interrupts */
195     SYS_INT_Enable();
196
197     /* Initialize the Application */
198     LED_BLINK_Initialize();
199 }
200
```

MHCで設定したSystem_serviceやDriverの初期化関数を呼び出している

SYS_Tasks関数

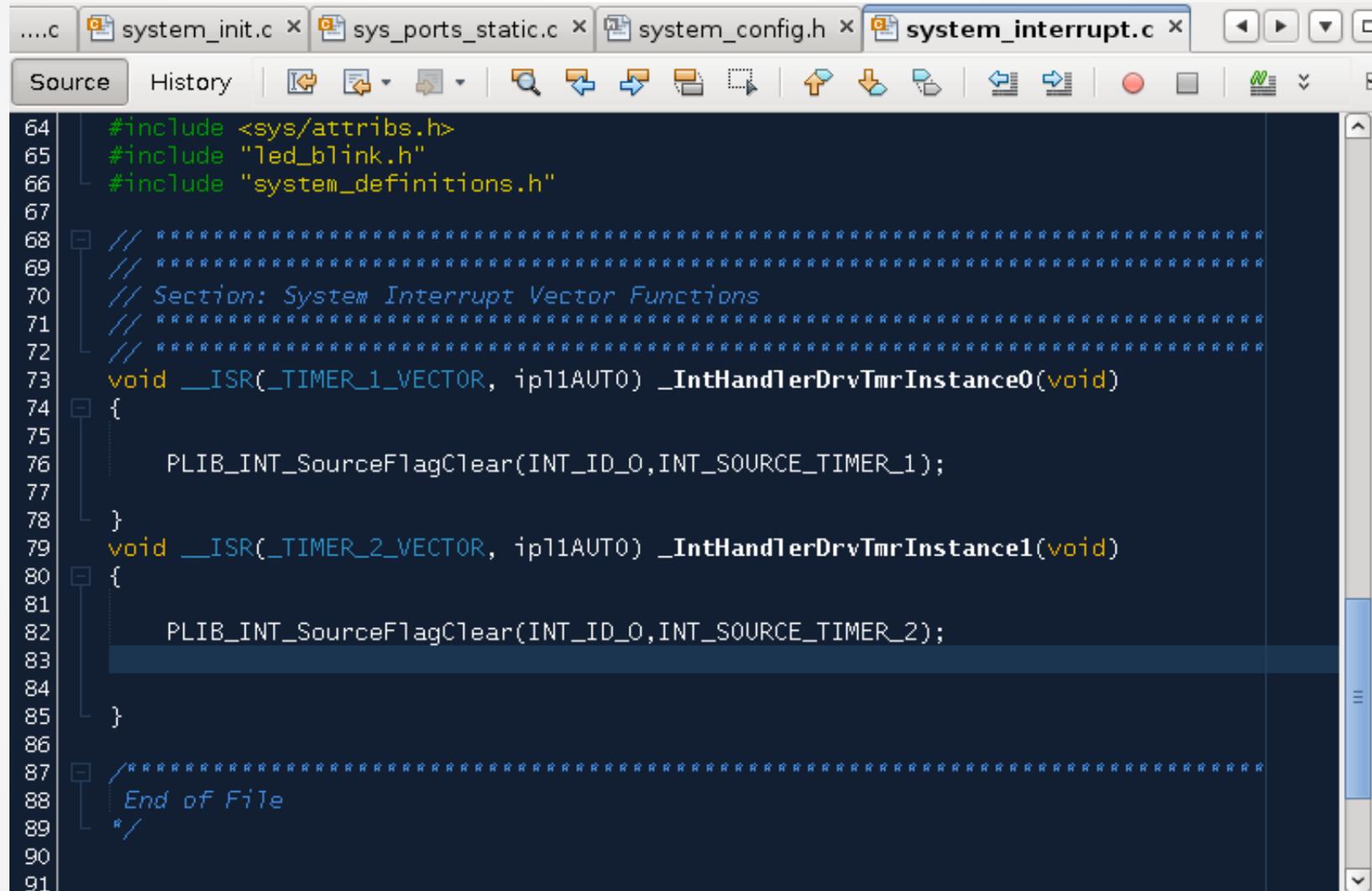


The screenshot displays the MPLAB X IDE interface. On the left, the 'Projects' pane shows the project structure for 'Led_Blinking'. The 'Source Files' folder is expanded, showing subfolders 'app', 'system_config', and 'framework'. The 'app' folder contains 'led_blink.c' and 'main.c'. The 'system_config' folder contains a 'default' subfolder with 'framework', 'driver', and 'system' subfolders. The 'framework' folder contains 'system_exceptions.c', 'system_init.c', 'system_interrupt.c', and 'system_tasks.c'. The 'system_tasks.c' file is selected, and its code is displayed in the main editor window. The code defines the `SYS_Tasks` function, which is responsible for maintaining system services, device drivers, and the application's state machine.

```
70      See prototype in system/common/sys_module.h.
71      */
72
73      void SYS_Tasks ( void )
74      {
75          /* Maintain system services */
76          SYS_DEVCON_Tasks(sysObj.sysDevcon);
77
78          /* Maintain Device Drivers */
79
80          /* Maintain Middleware & Other Libraries */
81
82
83          /* Maintain the application's state machine. */
84          LED_BLINK_Tasks();
85      }
86
87
```

SYS_DEVCON_Tasks (システム・レベルのステートマシン)は機能していない。

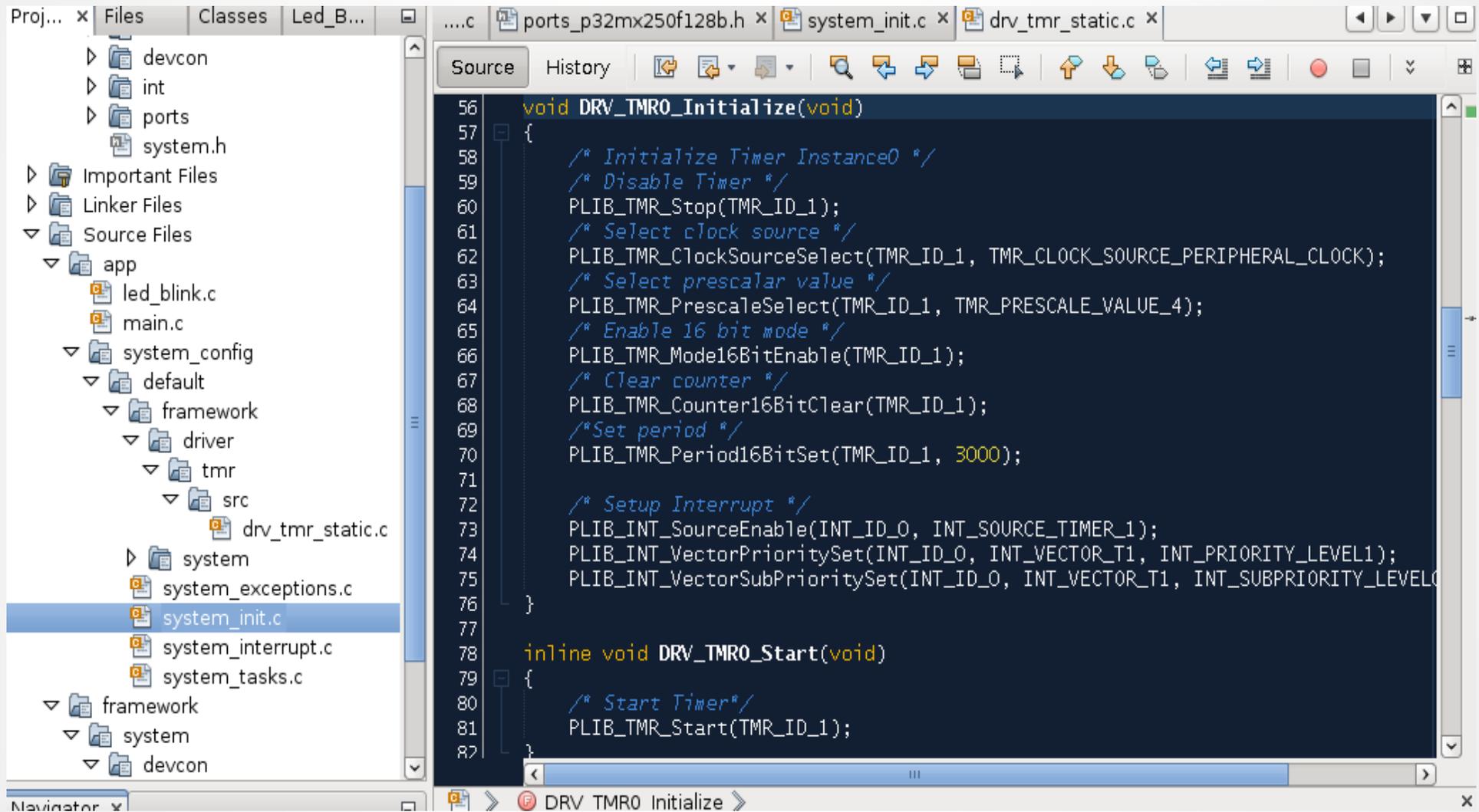
system_interrupt.c



```
64 #include <sys/attrs.h>
65 #include "led_blink.h"
66 #include "system_definitions.h"
67
68 // *****
69 // *****
70 // Section: System Interrupt Vector Functions
71 // *****
72 // *****
73 void __ISR(_TIMER_1_VECTOR, ip11AUTO) _IntHandlerDrvTmrInstance0(void)
74 {
75     PLIB_INT_SourceFlagClear(INT_ID_0,INT_SOURCE_TIMER_1);
76 }
77
78 void __ISR(_TIMER_2_VECTOR, ip11AUTO) _IntHandlerDrvTmrInstance1(void)
79 {
80     PLIB_INT_SourceFlagClear(INT_ID_0,INT_SOURCE_TIMER_2);
81 }
82
83
84
85 }
86
87 // *****
88 // End of File
89 //
90
91
```

Timer割り込みハンドラが自動生成される

DRV_TMR0_Initialize関数

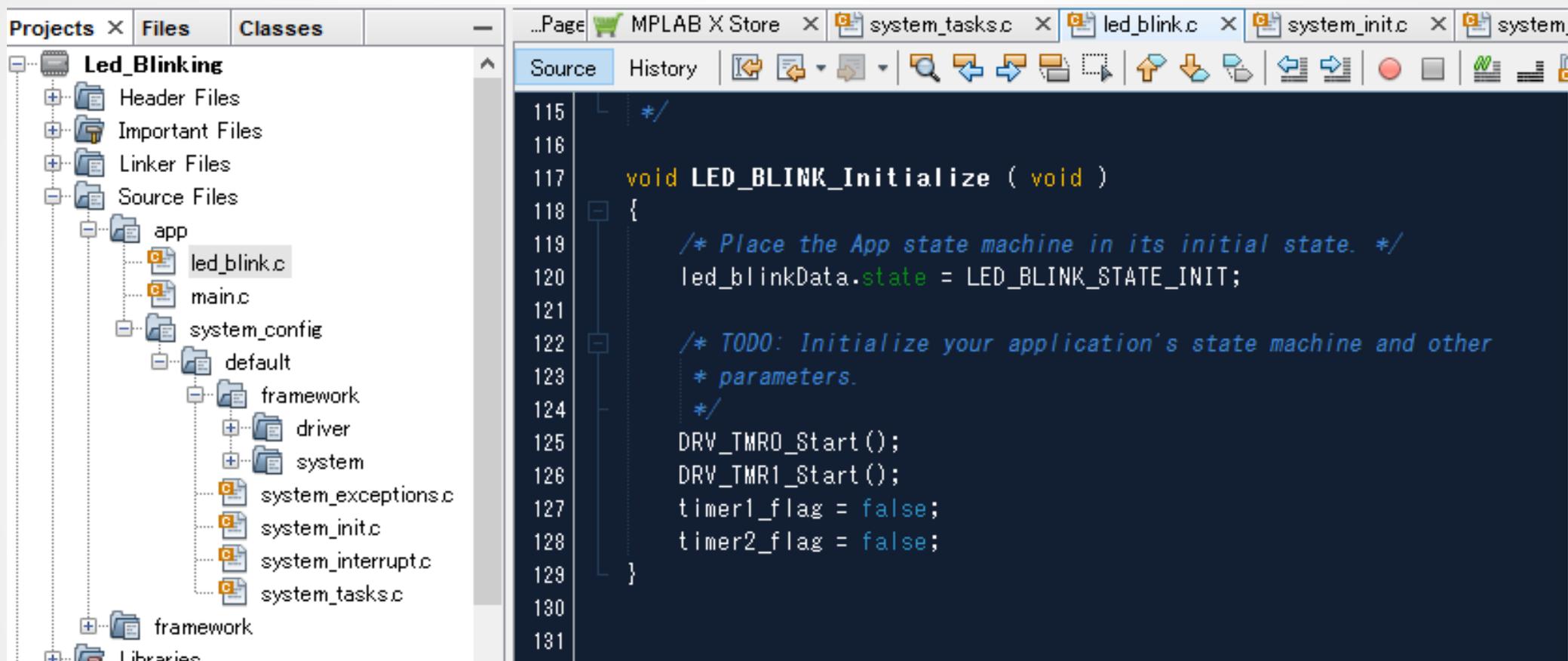


The screenshot shows an IDE window with a project explorer on the left and a code editor on the right. The project explorer shows a directory structure with 'system_init.c' selected. The code editor displays the implementation of the 'DRV_TMR0_Initialize' function, which configures a timer (TMR_ID_1) with various settings like clock source, prescaler, mode, and interrupt.

```
56 void DRV_TMR0_Initialize(void)
57 {
58     /* Initialize Timer Instance0 */
59     /* Disable Timer */
60     PLIB_TMR_Stop(TMR_ID_1);
61     /* Select clock source */
62     PLIB_TMR_ClockSourceSelect(TMR_ID_1, TMR_CLOCK_SOURCE_PERIPHERAL_CLOCK);
63     /* Select prescalar value */
64     PLIB_TMR_PrescaleSelect(TMR_ID_1, TMR_PRESCALE_VALUE_4);
65     /* Enable 16 bit mode */
66     PLIB_TMR_Mode16BitEnable(TMR_ID_1);
67     /* Clear counter */
68     PLIB_TMR_Counter16BitClear(TMR_ID_1);
69     /* Set period */
70     PLIB_TMR_Period16BitSet(TMR_ID_1, 3000);
71
72     /* Setup Interrupt */
73     PLIB_INT_SourceEnable(INT_ID_0, INT_SOURCE_TIMER_1);
74     PLIB_INT_VectorPrioritySet(INT_ID_0, INT_VECTOR_T1, INT_PRIORITY_LEVEL1);
75     PLIB_INT_VectorSubPrioritySet(INT_ID_0, INT_VECTOR_T1, INT_SUBPRIORITY_LEVEL0);
76 }
77
78 inline void DRV_TMR0_Start(void)
79 {
80     /* Start Timer*/
81     PLIB_TMR_Start(TMR_ID_1);
82 }
```

Timer Driver の設定に基づき初期化関数が生成される

LED_BLINK_Initialize関数



The screenshot displays the MPLAB X IDE interface. On the left, the Project Explorer shows the project structure for 'Led_Blinking'. The 'Source Files' folder is expanded, showing subfolders like 'app', 'system_config', and 'framework'. The 'app' folder contains 'led_blink.c' and 'main.c'. The 'system_config' folder contains a 'default' subfolder with 'framework', 'driver', and 'system' subfolders. The 'framework' folder contains 'system_exceptions.c', 'system_init.c', 'system_interrupt.c', and 'system_tasks.c'. The 'system_tasks.c' file is selected in the Project Explorer.

The main editor window shows the code for the 'LED_BLINK_Initialize' function in 'led_blink.c'. The code is as follows:

```
115  /*  
116  
117  void LED_BLINK_Initialize ( void )  
118  {  
119      /* Place the App state machine in its initial state. */  
120      led_blinkData.state = LED_BLINK_STATE_INIT;  
121  
122      /* TODO: Initialize your application's state machine and other  
123      * parameters.  
124      */  
125      DRV_TMRO_Start();  
126      DRV_TMR1_Start();  
127      timer1_flag = false;  
128      timer2_flag = false;  
129  }  
130  
131
```

Timer起動とFlagの初期化コードを追加する
(FlagはTimer割り込みハンドラでセットする)

LED_BLINK_Tasks関数

```
void LED_BLINK_Tasks ( void )
{
    switch ( led_blinkData.state )
    {
        case LED_BLINK_STATE_INIT:
            led_blinkData.state = LED_BLINK_STATE_RUN;
            loop_count = 0;
            break;
        case LED_BLINK_STATE_RUN:
            if( timer1_flag )
            {
                timer1_flag = false;
                if( ++loop_count >= 1000 )
                {
                    SYS_PORTS_PinToggle( PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_2);
                    loop_count = 0;
                }
            }
            if( timer2_flag )
            {
                timer2_flag = false;
                SYS_PORTS_PinToggle( PORTS_ID_0, PORT_CHANNEL_B, PORTS_BIT_POS_3);
            }
            break;
    }
}
```

これでLチカのプログラムが完成した

おまけ

```
123
124 void _general_exception_handler ( void )
125 {
126     /* Mask off Mask of the ExcCode Field from the Cause Register
127     Refer to the MIPS Software User's manual */
128     _except_code = (Address = 0xA0000204, _except_addr = 0x9D0008D4 >> 2);
129     _except_addr = _CPO_GET_EPC();
130     _cause_str = cause[_except_code];
131
132     SYS_DEBUG_PRINT(SYS_ERROR_ERROR, "%nGeneral Exception %s (cause=%d, addr=%x).%n",
133     _cause_str, _except_code, _except_addr);
134
135     while (1)
136     {
137         SYS_DEBUG_BreakPoint();
138     }
139 }
140
```

シミュレータで実行したらData bus errorを起こした

逆Asemmmble List

```
10820 71:          */
10821 72:
10822 73:          void SYS_Tasks ( void )
10823 74:          {
10824 9D0008D0 27BDFFE8 ADDIU SP, SP, -24
10825 9D0008D4 AFBF0014 SW RA, 20(SP)
10826 75:          /* Maintain system services */
10827 76:          SYS_DEVCON_Tasks(sysObj.sysDevcon);
10828 9D0008D8 3C02A000 LUI V0, -24576
10829 9D0008DC 0F400269 JAL SYS_DEVCON_Tasks
10830 9D0008E0 8C440220 LW A0, 544(V0)
10831 77:
10832 78:          /* Maintain Device Drivers */
10833 79:
```

SyS_tasks関数のレジスタ待避でスタック・オーバーフローを起こしている。実機では起きなかったもので、シミュレータのバグではないか (?) と思われる

おまけのおまけ

The screenshot shows an IDE interface with several windows and a menu open. A red arrow points to the 'disassembly' folder in the project tree. The 'Debugging' menu is open, showing a sub-menu 'Output' which contains 'Disassembly Listing File'. The 'Output' window displays assembly code for a PIC18F45K20 device, including a disassembly listing file.

Projects: Files × Classes

- Led_Blinking - C:\Users\akira\Desktop\Led_Blink\firmware
- Led_Blinking - C:\Users\akira\Desktop\Led_Blink\firmware
- Led_Blinking**
 - build
 - debug
 - disassembly
 - dist
 - nbproject
 - Makefile
 - disassemble
- PIC18F45K20

Window Help

- Projects Ctrl+1
- Files Ctrl+2
- Classes Ctrl+9
- Favorites Ctrl+3
- Services Ctrl+5
- Dashboard
- Navigator Ctrl+7
- Action Items Ctrl+6
- Tasks Ctrl+Shift+6
- Output Ctrl+4
- Editor Ctrl+0
- Debugging
 - Output
 - Disassembly Listing File
 - Variables Alt+Shift+1
 - Watches Alt+Shift+2
 - Call Stack Alt+Shift+3
 - Breakpoints Alt+Shift+5
 - Sessions Alt+Shift+6
 - Sources Alt+Shift+8
 - Disassembly
 - PIC AppIO
 - Trace
 - Stopwatch
 - PC Profiling
 - Triggers
 - Debugger Console
- Web
- IDE Tools
- PIC Memory Views
- Simulator
- Configure Window
- Reset Windows
- Close Window Ctrl+W
- Close All Documents Ctrl+Shift+W
- Close Other Documents
- Document Groups
- Documents... Shift+F4

PC: 0x9D000858 How do I? Keyword(s)

```
system_interrupt.c × main.c × sys_devcon.c × system_except
```

Code Field from the Cause Register
User's manual */
SE() & 0x0000007C) >> 2;
);
code];
ERROR, "%nGeneral Exception %s (cause=%d, addr=%x).%n",
_excep_code, _excep_addr);

general_exception_han... Led_Blinking - Dashb... ×

Led_Blinking
Project Type: Application - Configuration: default
Device
PIC32MX250F128B
Checksum: 0xFE000F87
Compiler Toolchain
XC32 (v1.40) [C:\Program Files (x86)\Microch
Production Image: Optimization: gcc O1 gcc++

Output ×