【SystemFactory.h】

```
namespace APPLI
{
	class SystemFactory
	{
	public:
		SystemFactory();
		virtual ~SystemFactory();
		void startup();
	};
}
```

【SystemFactory.cpp】

```
#include "SystemFactory.h"
#include  "EE_LIB/HAL/IO/IOPin.h"
#include  "EE_LIB/HAL/IntervalTimer/Timer8Bit.h"
#include  "EE_LIB/HAL/Interrupt/Priority.h"
#include "EE_LIB/OSWrapper/Task/OSTimer.h"
#include "EE_LIB/OSWrapper/Task/CyclicTask.h"

#include  "LED_Task.h"
using APPLI::SystemFactory;

SystemFactory::SystemFactory() {}
SystemFactory::~SystemFactory() {}

void SystemFactory::startup()
{
	EE_LIB::HAL::IO::IOPin p65(EE_LIB::HAL::IO::IOPORT6,EE_LIB::HAL::IO::IOPIN5,EE_LIB::HAL::IO::IOPin::DIR_OUT);
	EE_LIB::HAL::IntervalTimer::Timer8Bit timer;
	EE_LIB::OSWrapper::Task::OSTimer osTimer;
	LED_Task ledTaskHandler(&p65);
	EE_LIB::OSWrapper::Task::CyclicTask    ledTask(&ledTaskHandler,1000,EE_LIB::OSWrapper::Task::Task::TP_Middle);
	// 1 秒周期のタスク

	timer.init(EE_LIB::HAL::IntervalTimer::Timer8Bit::Timer0,1000000UL,EE_LIB::HAL::Interrupt::PRIORITY_LOW,&osTimer
);
	osTimer.addTask(&ledTask);

	while(-1)  {}
}
```

【LED_Task.h】

```
#include "EE_LIB/OSWrapper/Task/CyclicTaskPerformer.h"
#include  "EE_LIB/HAL/IO/IOPin.h"

namespace APPLI
{
		class LED_Task: public EE_LIB::OSWrapper::Task::CyclicTaskPerformer
		{

		public:
```

```
        LED_Task (EE_LIB::HAL::IO::IOPin* pin) ;
        virtual ~LED_Task () ;

        void perform (unsigned short interval) ;

    private:
        unsigned char m_state;
        EE_LIB::HAL::IO::IOPin* m_pin;
    } ;

}
```

【LED_Task.cpp】

```
#include "LED_Task.h"

using APPLI::LED_Task;

LED_Task::LED_Task (EE_LIB::HAL::IO::IOPin* pin) :m_state (0) ,m_pin (pin)  {}
LED_Task::~LED_Task ()  {}

void LED_Task::perform (unsigned short interval)
 {
        m_pin->set (m_state) ;
        m_state ^=1;
}
```