main.c

```
/**
  ******************************************************************************
  * @file    IO_Toggle/main.c
  * @author  MCD Application Team
  * @version V1.0.0
  * @date    23-March-2012
  * @brief   Main program body
  ******************************************************************************
  */

/* Includes ------------------------------------------------------------------*/
#include "stm32f0xx.h"


/* Private typedef -----------------------------------------------------------*/
/* Private define ------------------------------------------------------------*/
#define BSRR_VAL        0x0300

/* Private macro -------------------------------------------------------------*/
/* Private variables ---------------------------------------------------------*/
GPIO_InitTypeDef        GPIO_InitStructure;

/* Private function prototypes -----------------------------------------------*/
/* Private functions ---------------------------------------------------------*/
void delay (int a);


int main(void)
{
  /*!< At this stage the microcontroller clock setting is already
configured,
       this is done through SystemInit() function which is called from
startup
       file (startup_stm32f0xx.s) before to branch to application main.
       To reconfigure the default setting of SystemInit() function, refer
to
       system_stm32f0xx.c file
     */

  /* GPIOC Periph clock enable */
  RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOC, ENABLE);
  /*GPIOA Periph clock enable */
  RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOA,ENABLE);

  /* Configure PC8 and PC9 in output pushpull mode */
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_8 | GPIO_Pin_9;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_OUT;
  GPIO_InitStructure.GPIO_OType = GPIO_OType_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_InitStructure.GPIO_PuPd = GPIO_PuPd_NOPULL;
  GPIO_Init(GPIOC, &GPIO_InitStructure);

  /*Configure PA0 et PA1 comme input  Pull-up */
  GPIO_InitStructure.GPIO_Pin= GPIO_Pin_0 |GPIO_Pin_1;
  GPIO_InitStructure.GPIO_Mode= GPIO_Mode_IN;
  GPIO_InitStructure.GPIO_PuPd= GPIO_PuPd_DOWN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
```

```
      while (1) {
      if(GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_0))
            {
            GPIO_WriteBit(GPIOC,GPIO_Pin_8,Bit_SET);
            GPIO_WriteBit(GPIOC,GPIO_Pin_9,Bit_RESET);
            delay(2000000);
            GPIO_WriteBit(GPIOC,GPIO_Pin_8,Bit_RESET);
            GPIO_WriteBit(GPIOC,GPIO_Pin_9,Bit_SET);
            delay(2000000);
            }
      else
            {
        /* Set PC8 and PC9 */
            GPIOC->BSRR = BSRR_VAL;
            delay(500000);
        /* Reset PC8 and PC9 */
            GPIOC->BRR = BSRR_VAL;
            delay(500000);
            }
      }
}

void delay (int a)
{
      volatile int i,j;

      for (i=0 ; i < a ; i++)
      {
            j++;
      }

      return;
}
#ifdef  USE_FULL_ASSERT

/**
  * @brief  Reports the name of the source file and the source line number
  *         where the assert_param error has occurred.
  * @param  file: pointer to the source file name
  * @param  line: assert_param error line source number
  * @retval None
  */
void assert_failed(uint8_t* file, uint32_t line)
{
  /* User can add his own implementation to report the file name and line
number,
     ex: printf("Wrong parameters value: file %s on line %d\r\n", file,
line) */

  /* Infinite loop */
  while (1)
  {
  }
}
#endif


/******************* (C) COPYRIGHT STMicroelectronics *****END OF FILE****/
```

makeFile.c

```
#
#       !!!! Do NOT edit this makefile with an editor which replace tabs by
spaces !!!!
#
##############################################################################
###################
#
# On command line:
#
# make all = Create project
#
# make clean = Clean project files.
#
# To rebuild project do "make clean" and "make all".
#
# Included originally in the yagarto projects. Original Author : Michael
Fischer
# Modified to suit our purposes by Hussam Al-Hertani
# Use at your own risk!!!!!
##############################################################################
###################
# Start of default section
#
CCPREFIX = arm-none-eabi-
CC   = $(CCPREFIX)gcc
CP   = $(CCPREFIX)objcopy
AS   = $(CCPREFIX)gcc -x assembler-with-cpp
GDBTUI = $(CCPREFIX)gdbtui
HEX  = $(CP) -O ihex
BIN  = $(CP) -O binary -S
MCU  = cortex-m0

# List all C defines here
DDEFS = -DSTM32F0XX -DUSE_STDPERIPH_DRIVER
#
# Define project name and Ram/Flash mode here
PROJECT        = iotogglem0_wspl

# List C source files here
LIBSDIRS    = ../../STM32F0308-Discovery_FW_V1.0.1/Libraries
CORELIBDIR = $(LIBSDIRS)/CMSIS/Include
DEVDIR  = $(LIBSDIRS)/CMSIS/Device/ST/STM32F0xx
STMSPDDIR    = $(LIBSDIRS)/STM32F0xx_StdPeriph_Driver
STMSPSRCDDIR = $(STMSPDDIR)/src
STMSPINCDDIR = $(STMSPDDIR)/inc
DISCOVERY    = ../../STM32F0308-Discovery_FW_V1.0.1/Utilities/STM32F0-
Discovery

#list of src files to include in build process

SRC  = ./src/main.c
SRC += ./src/stm32f0xx_it.c
SRC += $(DEVDIR)/Source/Templates/system_stm32f0xx.c

## used parts of the STM-Library
#SRC += $(STMSPSRCDDIR)/stm32f0xx_adc.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_cec.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_crc.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_comp.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_dac.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_dbgmcu.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_dma.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_exti.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_flash.c
SRC += $(STMSPSRCDDIR)/stm32f0xx_gpio.c
```

```
#SRC += $(STMSPSRCDDIR)/stm32f0xx_syscfg.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_i2c.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_iwdg.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_pwr.c
SRC += $(STMSPSRCDDIR)/stm32f0xx_rcc.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_rtc.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_spi.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_tim.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_usart.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_wwdg.c
#SRC += $(STMSPSRCDDIR)/stm32f0xx_misc.c

# List assembly startup source file here
STARTUP = ./startup/startup_stm32f0xx.s

# List all directories here
INCDIRS = $(DEVDIR)/Include \
          $(CORELIBDIR) \
          $(STMSPINCDDIR) \
          $(DISCOVERY)    \
          ./inc
# List the user directory to look for the libraries here
LIBDIRS += $(LIBSDIRS)

# List all user libraries here
LIBS =

# Define optimisation level here
OPT = -Os


# Define linker script file here
LINKER_SCRIPT = ./linker/stm32f0_linker.ld


INCDIR  = $(patsubst %,-I%, $(INCDIRS))
LIBDIR  = $(patsubst %,-L%, $(LIBDIRS))
LIB     = $(patsubst %,-l%, $(LIBS))
##reference only flags for run from ram...not used here
##DEFS    = $(DDEFS) $(UDEFS) -DRUN_FROM_FLASH=0 -DVECT_TAB_SRAM

## run from Flash
DEFS    = $(DDEFS) -DRUN_FROM_FLASH=1

OBJS  = $(STARTUP:.s=.o) $(SRC:.c=.o)
MCFLAGS = -mcpu=$(MCU)

ASFLAGS = $(MCFLAGS) -g -gdwarf-2 -mthumb  -Wa,-amhls=$(<:.s=.lst)
CPFLAGS = $(MCFLAGS) $(OPT) -g -gdwarf-2 -mthumb   -fomit-frame-pointer -
Wall -Wstrict-prototypes -fverbose-asm -Wa,-ahlms=$(<:.c=.lst) $(DEFS)
LDFLAGS = $(MCFLAGS) -g -gdwarf-2 -mthumb -nostartfiles -T$(LINKER_SCRIPT)
-Wl,-Map=$(PROJECT).map,--cref,--no-warn-mismatch $(LIBDIR) $(LIB)

#
# makefile rules
#

all: $(OBJS) $(PROJECT).elf  $(PROJECT).hex $(PROJECT).bin
      $(TRGT)size $(PROJECT).elf

%o: %c
      $(CC) -c $(CPFLAGS) -I . $(INCDIR) $< -o $@

%o: %s
      $(AS) -c $(ASFLAGS) $< -o $@

%elf: $(OBJS)
```

```
	$(CC) $(OBJS) $(LDFLAGS) $(LIBS) -o $@

%hex: %elf
	$(HEX) $< $@

%bin: %elf
	$(BIN)  $< $@

flash_openocd: $(PROJECT).bin
	openocd -s ~/EmbeddedArm/openocd-bin/share/openocd/scripts/ -f
interface/stlink-v2.cfg -f target/stm32f0x_stlink.cfg -c "init" -c "reset
halt" -c "sleep 100" -c "wait_halt 2" -c "flash write_image erase
$(PROJECT).bin 0x08000000" -c "sleep 100" -c "verify_image $(PROJECT).bin
0x08000000" -c "sleep 100" -c "reset run" -c shutdown

flash_stlink: $(PROJECT).bin
	st-flash write $(PROJECT).bin 0x8000000

erase_openocd:
	openocd -s ~/EmbeddedArm/openocd-bin/share/openocd/scripts/ -f
interface/stlink-v2.cfg -f target/stm32f0x_stlink.cfg -c "init" -c "reset
halt" -c "sleep 100" -c "stm32f1x mass_erase 0" -c "sleep 100" -c shutdown

erase_stlink:
	st-flash erase

debug_openocd: $(PROJECT).elf flash_openocd
	xterm -e openocd -s ~/EmbeddedArm/openocd-bin/share/openocd/scripts/
-f interface/stlink-v2.cfg -f target/stm32f0x_stlink.cfg -c "init" -c
"halt" -c "reset halt" &
	$(GDBTUI) --eval-command="target remote localhost:3333"
$(PROJECT).elf

debug_stlink: $(PROJECT).elf
	xterm -e st-util &
	$(GDBTUI) --eval-command="target remote localhost:4242"
$(PROJECT).elf -ex 'load'

clean:
	-rm -rf $(OBJS)
	-rm -rf $(PROJECT).elf
	-rm -rf $(PROJECT).map
	-rm -rf $(PROJECT).hex
	-rm -rf $(PROJECT).bin
	-rm -rf $(SRC:.c=.lst)
	-rm -rf $(ASRC:.s=.lst)
# *** EOF ***
```