UNIVERSITA' DI SALERNO

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE ED ELETTRICA E MATEMATICA APPLICATA

Laurea Magistrale in Ingegneria Informatica

Project work

Deliverable 3

Sistemi Embedded

Gruppo: 8

Marotta Giuseppe - 0622702302 - g.marotta31@studenti.unisa.it

Rea Gaetano - 0622702190 - g.rea7@studenti.unisa.it

Squitieri Giuseppe - 0622702339 - g.squitieri8@studenti.unisa.it

Tramice Davide - 0622702194 - d.tramice@studenti.unisa.it

ANNO ACCADEMICO 2023/2024

Indice

1 Generazione del codice e implementazione

In questa sezione verranno presentati i passaggi attuati per effettuare la generazione del codice e il successivo deployment sulla scheda **STM-32 NUCLEO-G474RE**. Successivamente si è passato ad un intensa campagna di testing direttamente sull'hardware, i risultati sono riportati nell'ultima parte.

1.1 Il circuito

Per creare il circuito è stata utilizzata una breadboard che permette di effettuare collegamenti ordinati senza l'utilizzo della saldatrice a stagno. Si è iniziato con il connettere i vari componenti alla breadboard, poi si sono collegate le alimentazioni dei componenti ove necessario ed infine si è collegata la scheda tramite i suoi pin.
La configurazione dei pin può essere vista nella successiva tabella:

Variabile	**Pin**	**Configurazione**
`B1`	PC13	PULL-DOWN
`B2`	PC6	PULL-DOWN
`B3`	PC8	PULL-DOWN
`P1`	PC10	PULL-UP
`P2`	PC12	PULL-UP
`RED_LED`	PB5	-
`YELLOW_LED`	PB4	-
`GREEN_LED`	PB3	-

Tabella 1: Variabili utilizzate nel modello Stateflow

1.2 Generazione del codice (Embedded coder)

Una volta configurata la scheda ed aver accertato che tutti i componenti siano funzionanti si è passato alla generazione del codice tramite il tool **Embedded coder**. Questo tool permette di generare i file .h e il file .c che descrivono il modello descritto in Simulink.

- **Setting variabili** Il primo passaggio effettuato è stato quello che tutte le variabili fossero del tipo giusto, in particolare, le variabili di input e output son di tipo textitboolean, mentre le restanti che riguardano i tempi sono di tipo *double*.
- **Step time** Successivamente si è scelto lo step time a 0.1 secondi.
- **Parametri generazione** A questo punto è stata avviata la generata del codice per un processore **ARM** di tipo **Cortex-M**.La generazione produce diversi file, ma quelli che servono sono in particolare 3: *AutomaticGate.h, AutomaticGate.c e rtwtypes.h*.
- **Import codice** A questo punto si importano i tre file nel progetto configurato in **STM32CubeIDE** e si procede alla build del progetto per assicurarsi che i file siano senza errori.
- **Main** L'ultima parte da effettuare è quella di configurare il file *main.c*. Nel main sono stati inclusi i file .h importati in precedenza, successivamente sono state create due funzioni per la lettura e la scrittura degli input e degli output ed infine è stata definita la sequenza di azioni da eseguire nel while infinito. Nello snippet di codice successivo è mostrato tutto il codice inserito.

```
23 /* Private includes ----------
24 /* USER CODE BEGIN Includes */
25 #include "AutomaticGate.h"
26 #include "rtwtypes.h"
27 /* USER CODE END Includes */
```

Figura 1: Include of the system.

```
/* USER CODE BEGIN 0 */
static void AutomaticGate_read_inputs(){
    rtU.B1 = HAL_GPIO_ReadPin(B1_GPIO_Port, B1_Pin);
    rtU.B2 = HAL_GPIO_ReadPin(B2_GPIO_Port, B2_Pin);
    rtU.B3 = HAL_GPIO_ReadPin(B3_GPIO_Port, B3_Pin);

    rtU.P1 = !HAL_GPIO_ReadPin(P1_GPIO_Port, P1_Pin);
    rtU.P2 = !HAL_GPIO_ReadPin(P2_GPIO_Port, P2_Pin);
}

static void AutomaticGate_write_outputs(){
    HAL_GPIO_WritePin(Red_Led_GPIO_Port, Red_Led_Pin, rtY.Red_Led);
    HAL_GPIO_WritePin(Yellow_Led_GPIO_Port, Yellow_Led_Pin, rtY.Yellow_Led);
    HAL_GPIO_WritePin(Green_Led_GPIO_Port, Green_Led_Pin, rtY.Green_Led);
}

/* USER CODE END 0 */
```

Figura 2: Input and output of the system.

```
  /* USER CODE BEGIN 2 */
  AutomaticGate_initialize();
  /* USER CODE END 2 */

  /* Infinite loop */
  /* USER CODE BEGIN WHILE */
  while (1)
  {
    /* USER CODE END WHILE */

    /* USER CODE BEGIN 3 */
      uint32_t start, elapsed;
      start = HAL_GetTick();
      AutomaticGate_read_inputs();
      AutomaticGate_step();
      AutomaticGate_write_outputs();
      elapsed = HAL_GetTick() - start;
      HAL_Delay(100-elapsed);

  }
  /* USER CODE END 3 */
}

```

Figura 3: Testing.

1.3 Testing

Una volta esportato il codice sulla scheda **STM-32 NUCLEO-G474RE** c'è il bisogno di testare se tutto funzione, in particolare sono stati ripetuti tutti i test eseguiti sul modello, successivamente sono indicati i risultati

Test	Esito
Test 1	SUPERATO
Test 2	SUPERATO
Test 3	SUPERATO
Test 4	SUPERATO
Test 5	SUPERATO
Test 6	SUPERATO
Test 7	SUPERATO

Tabella 2: Variabili utilizzate nel modello Stateflow

Elenco delle figure