



**Politecnico
di Torino**

Model-based Software Design Assignment Report

S274816

Feihong Shi

Outline

Section1: Project Logic

Section2: Model Descriptions

Section3: Test Harness

Section4: Google Test

Section5: Trampoline

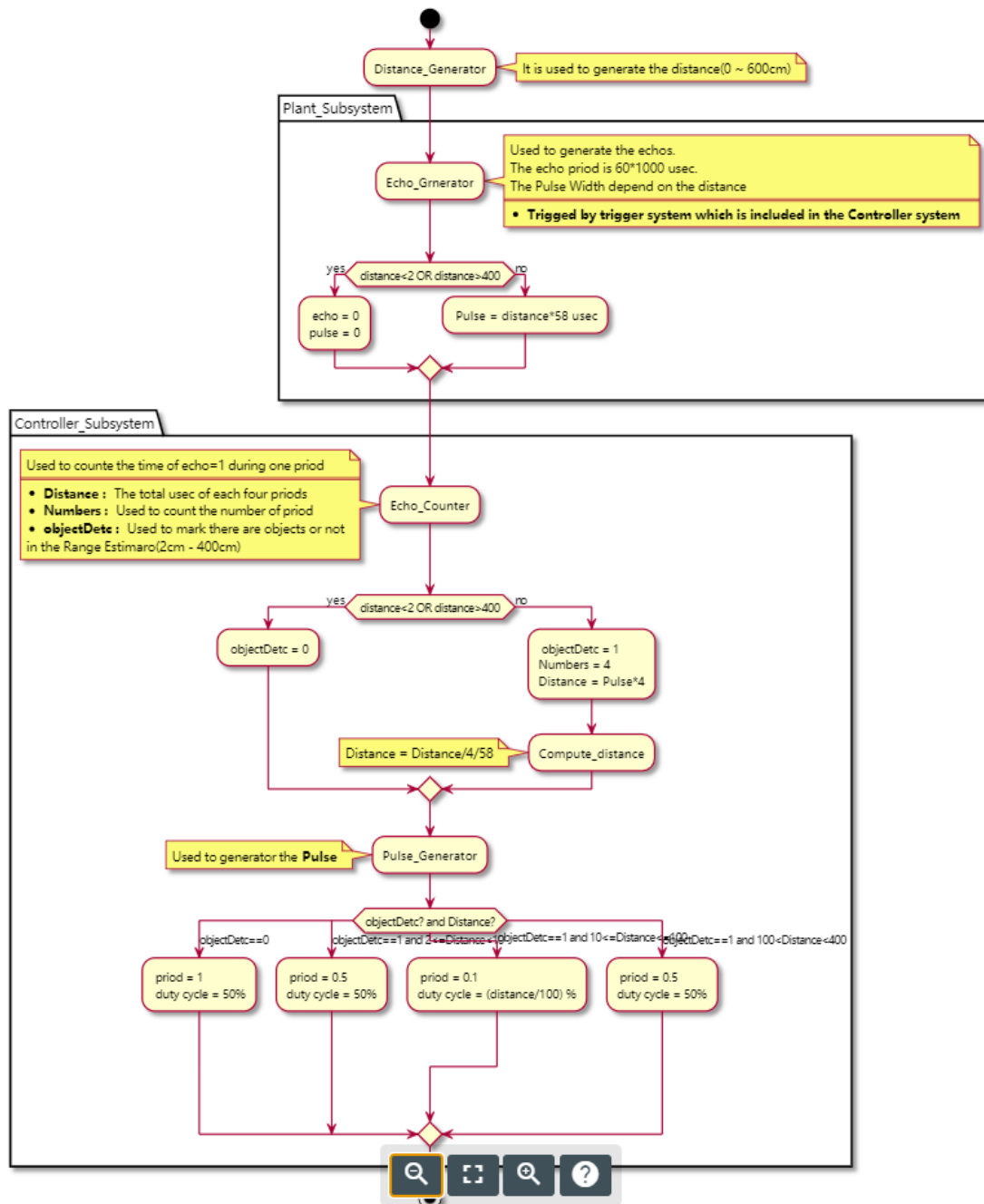
Section6: Arduino

Section1. Project Logic

Introduction

In this section, I will draw a flow char to explain my design logic.

Flow char



Section2. Project Logic

Introduction

There are two subsystem in my model, one is for generating the echo, another one is for generating the pulse.

The Plant subsystem has one stateflow, triggered each 60 msec, that is used to generate the echo, and the pulse width(echo = 1 time) is depend on the distance.

The Controller subsystem has three stateflows, one is for triggering the Plant subsystem, the second one is for counting the pulse width and compute the distance to check there are object or not, the last one is for generating the pulse for the each kind scenario.

Details

Plant Interfaces

Name	Direction	Data type	Descriptions
distance	input	single	The distance of the detecting object.
trigger	input	single	Generating by the controller subsystem each 60ms.
echo	output	single	Period = 60 ms, Pulse width depends on the distance.

Controller Interfaces

Name	Direction	Data type	Descriptions
echo	input	single	Period = 60 ms, generating by Plant subsystem, Pulse width depends on the distance.
trigger	output	single	Trigging the Plant subsystem each 60 ms to generator the echo
pulse	output	single	Generating the Pulse, the frequencies and period depend on the distance that computed by this controller subsystem.

Plant StateFlow

Name	Input	Output	Descriptions
Create_echos	Distance, trigger	echo	Trigging by trigger signal, generating the echo by distance signal.

Controller StateFlow

Name	Input	Output	Descriptions
Create_trigger		trigger	Generating the trigger signal each 60 ms to the plant subsystem
Echo_numbers1	echo	objectDetc, distance, numbers	Generating the Output signals by the input signal echo. objectDetc mark there is object or not. Distance represent the distance of the object. Numbers indicates the number of the distance that has been added
Create_pulses	objectDetc, distance, numbers	Pulse	Generating the pulse for the each kind scenario(different distances of object)

Section3: Test Harness

I create 7 test harness for the controller subsystem. Details are following in the table

Test Harness

Name	Period	Width Pulse	Test Sec	Distance	Pulse
RangingEstimatorV3Test1CM	60 ms	58 us	1	1 cm	1 Hz, duty cycle 50%
RangingEstimatorV3Test10CM	60 ms	580 us	0.3	10 cm	10 Hz, duty cycle 10%
RangingEstimatorV3Test50CM	60 ms	2900 us	0.3	50 cm	10 Hz, duty cycle 50%
RangingEstimatorV3Test100CM	60 ms	5800 us	0.3	100 cm	10 Hz, duty cycle 100%
RangingEstimatorV3Test200CM	60 ms	11600 us	0.7	200 cm	2 Hz, duty cycle 50%
RangingEstimatorV3Test400	60 ms	23200 us	0.7	400 cm	2 Hz, duty cycle 50%
RangingEstimatorV3Test600CM	60 ms	34800 us	1	600 cm	1 Hz, duty cycle 50%

Section4: Google Test

I have wrote the google test file(ert_main.c). Over all, The Lines Coverage is **92.3%**. The Lines Coverage is 88.9%. The following is the test result:

GCC Code Coverage Report

Directory: ./	Exec	Total	Coverage
Date: 2021-07-07 23:50:23	Lines: 250	271	92.3 %
Legend: low: < 75.0 % medium: >= 75.0 % high: >= 90.0 %	Branches: 139	269	51.7 %

File	Lines	Branches
Controller0.c	88.9 % 169 / 190	66.7 % 74 / 111
ert_main.c	100.0 % 81 / 81	41.1 % 65 / 158

Generated by: [GCOVR \(Version 3.4\)](#)

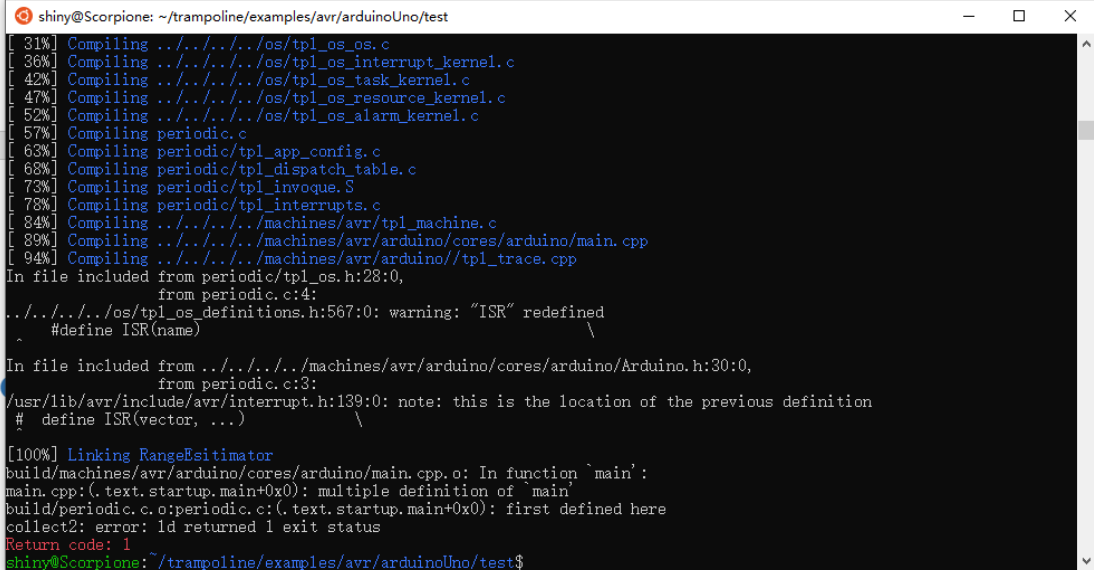
Section5: Trampoline

In this part of work, I successfully build the **periodic_exe** file and execute on virtual platform. The following is the building result:

```
BSWreadEcho(6): 1
BSWwriteTrig(6): 0
BSWwriteSignal (6): 0
---
BSWreadEcho(7): 1
BSWwriteTrig(7): 0
BSWwriteSignal (7): 0
---
BSWreadEcho(8): 1
BSWwriteTrig(8): 0
BSWwriteSignal (8): 0
---
BSWreadEcho(9): 1
BSWwriteTrig(9): 0
BSWwriteSignal (9): 0
Shutdown
Exiting virtual platform.
shiny@Scorpione: /trampoline/examples/posix/periodic$
```

Section6: Arduino

For this part, I just finished part of them, because of the ”./make.py” error I can not fix it.



```
shiny@Scorpione: ~/trampoline/examples/avr/arduinoUno/test
[ 31%] Compiling ../../../../os/tpl_os_os.c
[ 36%] Compiling ../../../../os/tpl_os_interrupt_kernel.c
[ 42%] Compiling ../../../../os/tpl_os_task_kernel.c
[ 47%] Compiling ../../../../os/tpl_os_resource_kernel.c
[ 52%] Compiling ../../../../os/tpl_os_alarm_kernel.c
[ 57%] Compiling periodic.c
[ 63%] Compiling periodic/tpl_app_config.c
[ 68%] Compiling periodic/tpl_dispatch_table.c
[ 73%] Compiling periodic/tpl_invoke.S
[ 78%] Compiling periodic/tpl_interrupts.c
[ 84%] Compiling ../../../../machines/avr/tpl_machine.c
[ 89%] Compiling ../../../../machines/avr/arduino/cores/arduino/main.cpp
[ 94%] Compiling ../../../../machines/avr/arduino/tpl_trace.cpp
In file included from periodic/tpl_os.h:28:0,
                 from periodic.c:4:
../../../../os/tpl_os_definitions.h:567:0: warning: "ISR" redefined
# define ISR(name)
^
In file included from ../../../../machines/avr/arduino/cores/arduino/Arduino.h:30:0,
                 from periodic.c:3:
/usr/lib/avr/include/avr/interrupt.h:139:0: note: this is the location of the previous definition
# define ISR(vector, ...)
^
[100%] Linking RangeEstimator
build/machines/avr/arduino/cores/arduino/main.cpp.o: In function `main':
main.cpp:(.text.startup.main+0x0): multiple definition of `main'
build/periodic.c.o:periodic.c:(.text.startup.main+0x0): first defined here
collect2: error: ld returned 1 exit status
Return code: 1
shiny@Scorpione: ~/trampoline/examples/avr/arduinoUno/test$
```