Introduction to Cyber Systems
Assignment 1 - Implementation of a FSMD simulator in Python

Benjy Jepsen
João Ramiro

September 2018

1 Introduction

The task was to create a simulator for Finite State Machines with Datapaths (FSMD) in python. The simulator takes in .xml files that describe the FSMD, and provide the stimuli for the FSMD while it is running. Two test cases to simulate FSMDs were provided: test_1 and test_2. In addition, we created our own test case, test_3.

2 Simulator

Code to parse the .xml documents into python-readable code, as well as helper code to evaluate conditions, set inputs, execute instructions, execute commands, inject stimuli, and check the endstate of the FSMD was provided. The code that was written to simulate an FSMD is relatively simple as it mostly relies on this helper code.

The main loop of the code iterates over the amount of times the FSMD transitions from state to state, or cycles. Stimuli are injected based on the current cycle of the FSMD, so the first element of the loop consists of the helper code that injects stimuli if there are any for the current cycle. Next, helper code is used to check if the endstate of the FSMD has been reached. If the endstate has been reached, the loop breaks. The current state, cycle, and variables are printed to provide information about how the FSMD is progressing.

The conditions of the current state are evaluated; the instruction associated with the condition that is true is executed; and the next state determined by the condition is reached. The loop begins again at this next state.

One important note: in case 2 or more conditions in the same state are true at the same time, the simulator will transition to the state for which the first condition is true, ignoring all further true conditions. This is done by inserting a break after executing the instructions associated with the first true condition. An alternative to this would be to print out a warning, telling the user that two or more conditions are true at the same time. Since it was only necessary to implement a simple, working simulator that does not check whether or not the state machine is implemented correctly, it was decided that transitioning to the first state for which a condition returns true is sufficient.

3 Test 1 and Test 2

Test 1 and Test 2 were provided as tests to determine if the FSMD simulator runs correctly.

Test 1 is a Mealy FSMD (the instructions are executed between states) that either increases or decreases the value of a variable, var_A, until it reaches the value of another threshold variable, var_TH. Test 1 does not include any stimuli, nor does it have an endstate.

Test 2 is a Moore FSMD (the instructions are executed in the states) that calculates the greatest common denominator of two values, in_A and in_B. Test 2 includes a stimuli .xml file and has an endstate.

4 Test 3

We had to provide the the third FSMD to test in the simulator. This test is a Mealy implementation of a missile launching mechanism. There are 3 inputs for this state machine: password, key 1 , and key 2. The steps to successfully launch a missile are as follows:

1. The secret password that opens the missile hatch must be inputted;
2. The secret password needed to reach the missile launching sequence must be inputted;
3. The two keys (key 1 and key 2) must be in a turned position at the same time within a 10 cycle period to trigger a missile launch;
4. A launch countdown lasts 10 cycles;
5. A Missile is launched;

First, a transition table (Table 1) was created to indicate the possible transitions between states; the conditions by which they are triggered; and the instructions that are executed during these transitions. For more information on CONDITIONs and INSTRUCTIONs , refer to the test3_desc.xml attached to this report.

STATES	NEXTSTATE	CONDITION	OPERATIONS
INITIALIZE	CHECK_PASSWORD	TRUE	SET_PASSWORD_HATCH SET_PASSWORD_LAUNCH RESET_TIMER_LAUNCH RESET_TIMER_KEY
CHECK_PASSWORD	CHECK_PASSWORD	PASSWORDS_INVALID	NOP
	CHECK_KEYS	LAUNCH_PASSWORD_VALID	REGISTER_PASSWORD_LAUNCH
	CHECK_KEYS	HATCH_PASSWORD_VALID	REGISTER_PASSWORD_HATCH
	CHECK_PASSWORD	PASSWORDS_INVALID	NOP
CHECK_KEYS	CHECK_KEYS	KEYS_WRONG	DECREASE_KEY_TIMER
	CHECK_PASSWORD	KEYS_NOT_TURNED_IN_TIME	RESET_HATCH_PASSWORD RESET_LAUNCH_PASSWORD RESET_TIMER_KEY
	INITIALIZE	KEYS_TURNED_IN_TIME_HATCH_PASSWORD_GOOD	OPEN_HATCH
	LAUNCH	KEYS_TURNED_IN_TIME_HATCH_OPEN_LAUNCH_PW_GOOD	NOP
LAUNCH	INITIALIZE	LAUNCH_TIMER_ZERO	LAUNCH CLOSE_HATCH SET_LAUNCH_PASSWORD_FALSE SET_HATCH_PASSWORD_FALSE RESET_TIMER_LAUNCH
	LAUNCH	LAUNCH_TIMER_NOT_ZERO	DECREASE_LAUNCH_TIMER

Table 1: Transition table of test_3

A state diagram was created to better visualize how the machine actually works (figure 1)

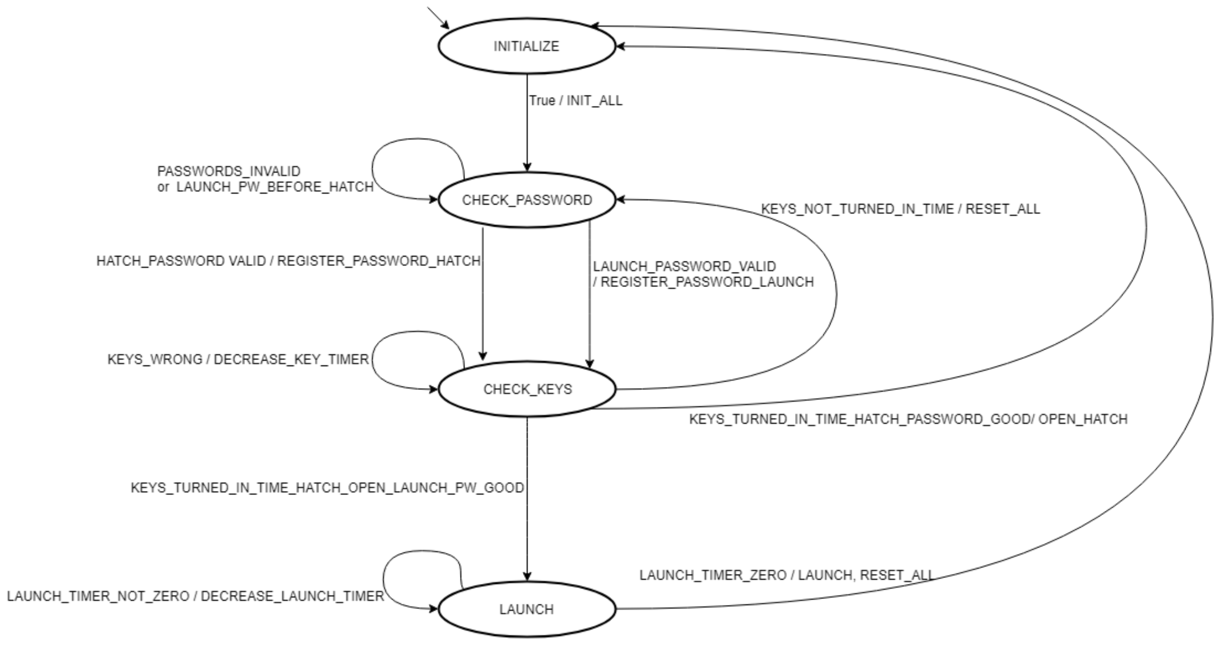
INITIALIZE
True / INIT_ALL
PASSWORDS_INVALID
or LAUNCH_PW_BEFORE_HATCH
CHECK_PASSWORD
KEYS_NOT_TURNED_IN_TIME / RESET_ALL
HATCH_PASSWORD VALID / REGISTER_PASSWORD_HATCH
LAUNCH_PASSWORD_VALID
/ REGISTER_PASSWORD_LAUNCH
KEYS_WRONG / DECREASE_KEY_TIMER
CHECK_KEYS
KEYS_TURNED_IN_TIME_HATCH_PASSWORD_GOOD/ OPEN_HATCH
KEYS_TURNED_IN_TIME_HATCH_OPEN_LAUNCH_PW_GOOD
LAUNCH_TIMER_ZERO / LAUNCH, RESET_ALL
LAUNCH_TIMER_NOT_ZERO / DECREASE_LAUNCH_TIMER
LAUNCH

Figure 1: Flowchart Diagram

A stimuli file for test 3 was also created. To check the robustness of the state machine, the following incorrect sequences of inputs were given as stimuli:

- Turning the keys without inputting the passwords.
- Inputting the launch password before inputting the password that opens the hatch.

The machine responded, correctly, to both sequences by not launching any missiles. Missiles were only launched with the correct sequence of inputs.

5 Conclusions

This assignment was a good introduction to understanding FSMDs. Although the assignment may seem daunting at first when looking through the .xml files and the python file, once the main concepts of how the FSMD should operate are understood, the assignment becomes relatively straight-forward. Creating our own test case turned out to be one of the more interesting parts of the assignment as we decided to test a relatively complicated FSMD.