Assignment 1
Part I

Name: Wubin Ouyang Student I.D:100934089

A description of Ultrasonic ranging system in DEVS model

In this assignment, I am going to model an Ultrasonic system in DEVS model. The system has 2 submodels. One is atomic model while another is a structural model which can be divided to four atomic models.The ranging system can detect the distance between an object and itself in a restricted range. When the receiver detects the echo, which also means that the distance is in the valid range, the system will output the distance by float-point number. The system diagram is shown in Figure-1.

As you can see the system consists of two submodels, the Receiver, which is an atomic model, and the ttp which further consists of three submodels. The Receiver is in charge of receiving echo, and ttp is responsible for transmitting and displaying the outcome. Here are descriptions of each atomic models.

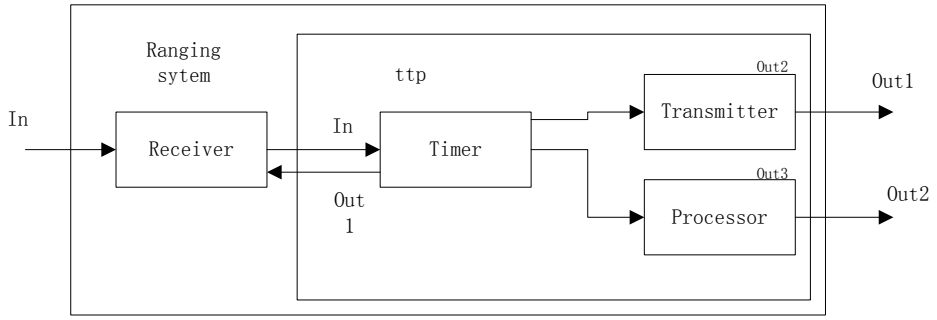


Figure-1 Systematic structure.

Receiver: It is enabled by the timer to be ready for echo. When it detects the echo, it will output a notification to the timer.

Timer: The timer has two different modes. When it enters the state of “waiting”, it enables the Receiver and Transmitter, disables the Processor and also begins timing. After an input from Receiver, it enters the “computing”. Simultaneously, it disables the receiver and gives the Processor data of time eclipsed to calculate the distance and display it. After 2 seconds, it automatically enters the state of “waiting” and begins a new cycle.

Transmitter: When it is enabled, it outputs the ultrasound.

Processor: It receives the data of time eclipsed after transmitting and does the calculation of distance. After the

calculation, the outcome will be output by it.

Part II

In this section, components of the model of Ultrasonic ranging system will be formed in the DEVS formalism.

Atomic MyReceiver=< X, Y, S, δint, δext, λ, ta>

Where:

X= Input Set= {(In1, 1); (In2, (0,1)) ;};

Y= Output= {(out, 1)};

S= State= {“ready”, “receiving”,”disable”};

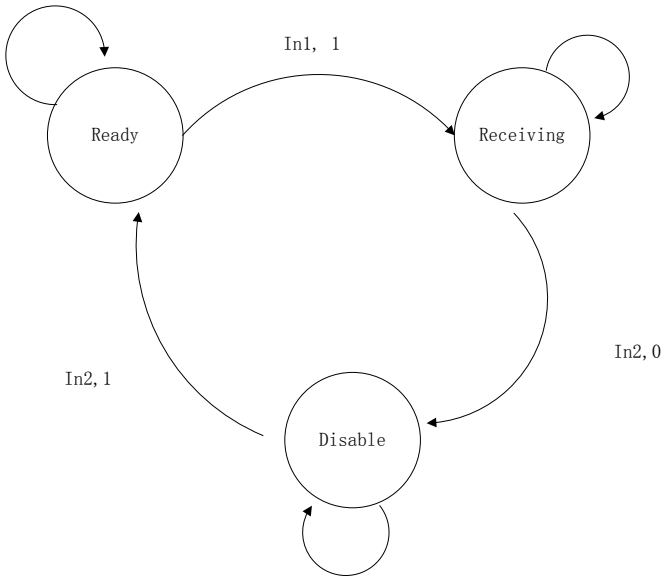
δext(“ready”, (In1,1))= “receiving”;

δext(“ready”, (Enable, 0))= “disable”;

δext(“disable”, (Enable, 1))= “ready”;

λ(“receiving”, e)= “1”;

ta=infinite;



Atomic MyTimer=< X, Y, S, δint, δext, λ, ta>

Where:

X= Input Set= {(In, 1)};

Y= Output= {(Out1, (0, 1)); (Out2,(0,1)); (Out3, (0,1))};

S= State= {“Waiting”, “Computing”};

δint(“Computing”)= “Waiting”;

δint(“Waiting”)= “Waiting”;

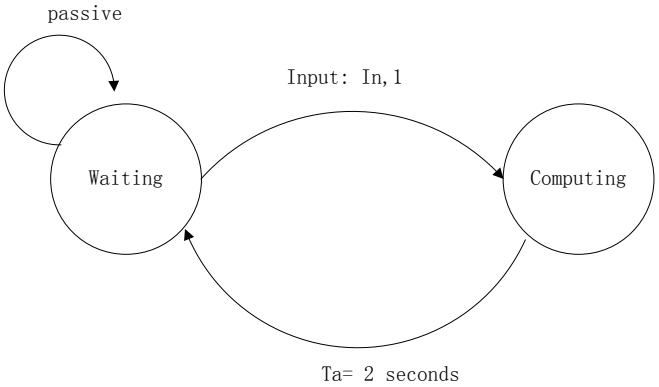
δext(“Waiting”, (In, 1))= “Computing”

λ(“Waiting”)= { (Out1,1), (Out2, 1), (Out3,0)};

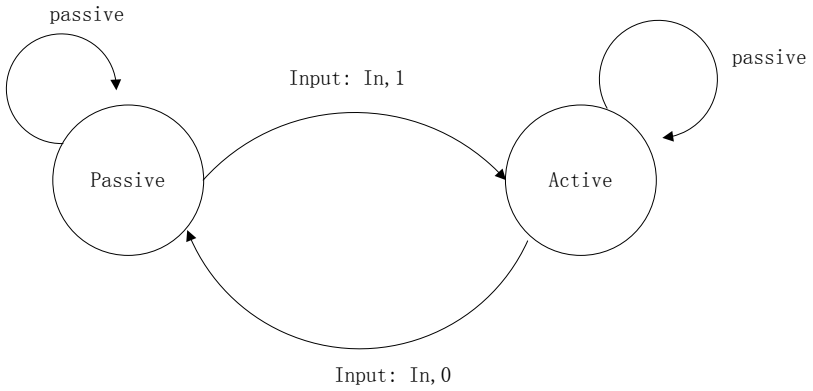
λ(“Computing”)= { (Out1,0), (Out2,0), (Out3,N+)};

ta(“Computing”)=2;

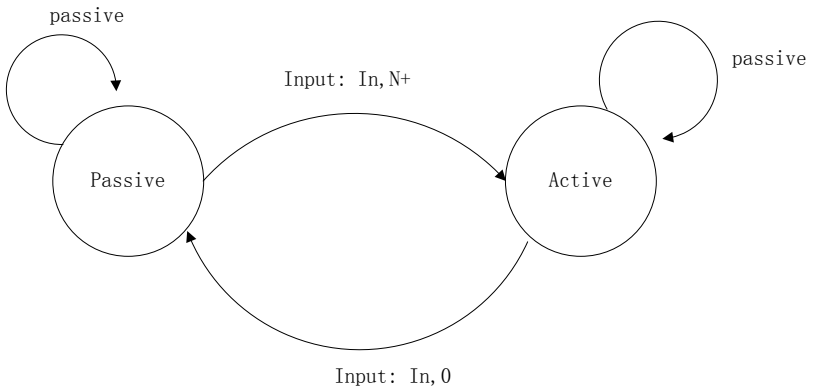
ta(“Waiting”)=1;



Atomic MyTransmitter =< X, Y, S, δint, δext, λ, ta>
Where:
X= Input Set= {(In,(0,1)) };
Y= Output= {(Out, (0,1))};
S= State= {“active”, “passive”};
δext(“passive”, (In,1)) =”active”;
δext(“active”, (In,0)) =”passive”;
λ(“active”) = 1
λ(“passive”) = 0
ta =infinite.



Atomic MyProcessor =< X, Y, S, δint, δext, λ, ta>
Where:
X= Input Set= {(In, N)};
Y= Output= {(Out, (0,1))};
S= State= {“active”, “passive”};
δext(“passive”, (In, N+)) =”active”;
δext(“active”, (In,0)) =”passive”;
λ(“active”) = 1
λ(“passive”) = 0
ta =infinite.



Coupled ttp <X, Y, D, {M_d|d∈D}, EIC, EOC, IC, select>

Where:

X= Input Set= {(In, 1)};

Y= Output = {(Out1, (0,1)), (Out2, N)};

D= { MyTimer, MyTransmitter, MyProcessor};

Md= { $M_{MyTimer}$, $M_{MyTransmitter}$, $M_{MyProcessor}$ };

EIC∈{((Self, In),(MyTimer, In)); }

EOC∈{((MyTimer, Out1),(Self, Out1)), ((MyTransmitter, Out),(Self,Out2));((MyProcessor, Out),(Self, Out3))}

IC∈{ ((MyTimer,Out2),(Transmitter, In));((MyTimer, Out3),(Processor, In))}

select={ MyTimer, MyTransmitter, MyProcessor};

Coupled Ranging Model <X, Y, D, {M_d|d∈D}, EIC, EOC, IC, select>

Where:

X= Input Set= {(In, 1)};

Y= Output = {(Out1, (0,1)), (Out2, N)};

D= {MyReceiver, ttp};

M_d= {$M_{Receiver}$, M_{ttp}};

EIC∈{((Self, In),(MyReceiver, In1)); }

EOC∈{((ttp, Out2),(Self, Out1));((ttp, Out3),(Self, Out2))}

IC∈{((MyReceiver, Out),(ttp, In)); }

select={MyReceiver, ttp};

Test Strategies

For each atomic and coupled models, different inputs will be provided to check if the outcome matches the expectancy. The order for the test is shown as below:

1. Test the “MyReceiver” atomic model;
2. Test the “MyTimer” atomic model;
3. Test the “MyTransmitter” atomic model;
4. Test the “MyProcessor” atomic model;
5. Test the “ttp” coupled model;
6. Test the “System” coupled model;

Part III

In this section, I will show the outcome of tests of each atomic model and the system model. The outcome proves that the system though it is in simulation can work properly.

1. MyReceiver

Model file:ReceiverTest.ma

Event file:ReceiverTest.ev

Description: When there is an input of value of “1” at port “In1”, the receiver will change its state from ready to receiving, and generate an output “1” at the port “Out”. When there is an input of value of 0 at port “In2”, the receiver will change its state from receiving to disable. Then any input at port “In1” will be ignored. After the input of value of “1” at port “In2”, the receiver will change its state from disable to ready and wait for the input at port “In1” again.

These are events for test.

```
00:00:00:05 In1 1
00:00:05:10 In2 0
00:00:07:05 In1 1
00:00:10:25 In2 1
00:00:15:40 In1 1
00:00:20:60 In2 0
00:00:25:80 In2 1
00:00:30:55 In1 1
00:00:35:15 In2 0
00:00:40:35 In2 1
00:00:45:15 In1 1
00:00:50:35 In2 0
00:00:55:50 In2 1
```

The outcome is show below:

```
00:00:00:005 out 1
00:00:15:040 out 1
00:00:30:055 out 1
00:00:45:015 out 1
```

The outcome proves the function of MyReceiver is correct. Notice the third line of events, this input did not generate a output at all because the MyReceiver is in the state of disable.

2. MyTimer

Timer:

Model: MyTimer

Model file:TimerTest.ma

Event file:TimerTest.ev

Description: When there is an input of value of “1” at port “In”, the Timer will change its state from waiting to computing, and generate an output “0” at the port “Out1” to disable the Receiver, and outputs “0” at “Out2” and a time eclipsed at “Out3” to stop transmit and display distance. After two seconds, Timer changes its state from computing to waiting, and generate an output “1” at the port “Out1” to enable the Receiver, and outputs “1” at “Out2” and “0” at “Out3” to transmit and disable the Processor.

These are events for the test:

```
00:00:05:05 In 1
00:00:07:10 In 1
```

The outcome is shown below:

```
00:00:01:000 out1 1
00:00:01:000 out2 1
00:00:01:000 out3 0
00:00:05:005 out1 0
00:00:05:005 out2 0
00:00:05:005 out3 5
00:00:07:005 out1 1
00:00:07:005 out2 1
00:00:07:005 out3 0
00:00:07:010 out1 0
00:00:07:010 out2 0
00:00:07:010 out3 1
```

The Timer entered the state of “waiting”, which is shown in Line1. And then it received a notification and entered the state of “computing” at 00:00:05:005, which is followed by an output of time eclipsed during “waiting” at out3. After 2 seconds, at 00:00:07:005 out1 1, it entered the state of “waiting”,

3. Transmitter

Model: MyTransmitter

Model file:MyTransmitter.ma

Event file:MyTransmitter.ev

Description: When there is an input of value of “1” at port “In”, the Transmitter will change its state from passive to active, and generate an output “1” at the port “Out” to transmit the ultrasound. When there is an input of value of “0” at port “In”, the Transmitter will change its state from active to passive, and generate an output “0” at the port “Out” to stop transmitting.

These are events for the test:

```
00:00:00:05 In 1
00:00:05:10 In 0
00:00:10:25 In 1
00:00:15:40 In 1
00:00:20:60 In 0
00:00:25:80 In 1
```

The outcome is shown as below:

```
00:00:00:005 out 1
00:00:05:010 out 0
00:00:10:025 out 1
00:00:20:060 out 0
00:00:25:080 out 1
```

The model is simple so that it can be proved correct simply. Actually it is just a symbol of an electrical device that can make and transmit ultrasound wave. To make the system more similar to a real model, I add this model though it just "pass" the input to the output.

4. Processor

Model: MyProcessor

Model file:MyProcessor.ma

Event file:MyProcessor.ev

Description: When there is an input of value that is not equal to "0" at port "In", the Processor will change its state from passive to active, and generate an output of a calculated distance at the port "Out" to display the distance. When there is an input of value of "0" at port "In", the Transmitter will change its state from active to passive, and generate an output "0" at the port "Out".

These are events for the test:

```
00:00:00:05 In 1
00:00:05:10 In 0
00:00:10:25 In 2
00:00:15:40 In 1
00:00:20:60 In 0
00:00:25:80 In 13
00:00:30:55 In 1
00:00:35:15 In 0
00:00:40:35 In 32
```

The outcome is shown as below:

```
00:00:00:005 out 2
00:00:05:010 out 0
00:00:10:025 out 4
00:00:20:060 out 0
00:00:25:080 out 26
00:00:35:015 out 0
00:00:40:035 out 64
```

The Processor receives the value of time eclipsed since transmitting ultrasound, and calculates and display the distance based on the value received. This model shows a simulated process. Notice at `00:00:15:40` and `00:00:30:55`, inputs are invalid. This because the Process had not been set to passive yet and could not receive any new input.

5. System

Model: System

Model file:SystemTest.ma

Event file:SystemTest.ev

Description: The system receives "1" at port "In". Then it outputs "0" at "Out1" and "1" at "Out2". After two seconds, it outputs "1" at "Out1" and "0" at "Out2" and waits for next "1" at port "In".

These are events for the test:

```
00:00:05:10 In 1
00:00:10:25 In 1
00:00:15:40 In 1
00:00:20:60 In 1
00:00:25:80 In 1
00:00:30:55 In 1
00:00:35:15 In 1
00:00:40:35 In 1
00:00:45:15 In 1
```

The outcome is shown as below:

```
00:00:02:005 out1 1
00:00:05:010 out2 8
00:00:05:010 out1 0
00:00:07:010 out2 0
00:00:07:010 out1 1
00:00:10:025 out2 8
00:00:10:025 out1 0
00:00:12:025 out2 0
00:00:12:025 out1 1
00:00:15:040 out2 8
00:00:15:040 out1 0
00:00:17:040 out2 0
00:00:17:040 out1 1
00:00:20:060 out2 8
00:00:20:060 out1 0
00:00:22:060 out2 0
00:00:22:060 out1 1
00:00:25:080 out2 8
00:00:25:080 out1 0
00:00:27:080 out2 0
00:00:27:080 out1 1
00:00:30:055 out2 6
00:00:30:055 out1 0
```

The System received “1” at `00:00:10:25` and then it output the distance “8” at out2 and disabled the transmitter with an output “0” at out1. After 2 seconds, at `00:00:12:025`, it enabled the transmitter with an output “1” at out1 and display “0” at out2.

Conclusion

By the test shown before, this ranging system has achieved a basic design for its functions. Though it still needs a more complicated mechanism to perform the real function of transmitting and calculating, it has proved that the architecture of model is fit for the implementation.

Some updates are still on the way to make this model better.