KIT-Fakultät für Informatik
Institut für Technische Informatik

Tutorial – ASIPmeister Adding Resources

Customized Embedded Processor Design

Application Specific Instruction-Set Processors- ASIP
Lab (Prakitikum)

Responsible/Author:
MSc. Sajjad Hussain

Supervisors:
MSc. Sajjad Hussain, Dr.-Ing. Lars Bauer, Prof. Dr.-Ing. Jörg Henkel

Chair of Embedded Systems,
Building 07.21, Haid-und-Neu-Str. 7,
76131 Karlsruhe, Germany.

November 2, 2023

ASIPMEISTER -ADDING NEW RESOURCE -TUTORIAL

DEFINE, TEST and Add Hardware

A. SET THE DIRECTORIES:

1. Login to any *i83labpcXX.itec.kit.edu* directly or using SSH or using X2Go Client. For example, login as *asip-sajjad04* into *i83labpc02.itec.kit.edu*

2. Open shell terminal from the start menu. It should be in your default home directory. Type "*pwd*"

```
sajjad@i83pc57:~:$pwd
/home/sajjad
```

3. Create a new directory for your Lab e.g. "*SS21*"

```
sajjad@i83pc57:~:$mkdir SS21
sajjad@i83pc57:~:$cd SS21/
```

4. Create a directory for your ASIP project in "*SS21*"

```
sajjad@i83pc57:~/SS21:$mkdir ASIPMeisterProjects
sajjad@i83pc57:~/SS21:$cd ASIPMeisterProjects/
```

5. Create a new directory for your lab session in "*ASIPMeisterProjects*" e.g. "*8*"

```
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects:$mkdir 8
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects:$cd 8/
```

B. IMPLEMENT A NEW RESOURCE AND TEST:

6. Write a new VHDL file of the required resource, for example, **MINMAX** to calculate the minimum and maximum from given two numbers.

```
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects/8:$vim minmax.vhd
```

Listing 1: "minmax.vhd"

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

use IEEE.std_logic_arith.all;
entity minmax is
Port ( clock : in std_logic; reset: in std_logic; enb: in std_logic;
din1 : in  STD_LOGIC_VECTOR (31 downto 0);
din2   : in  STD_LOGIC_VECTOR (31 downto 0);
```

```vhdl
doutMin   : out  STD_LOGIC_VECTOR (31 downto 0);
doutMax   : out STD_LOGIC_VECTOR (31 downto 0)
);
end minmax;

architecture st of minmax is
begin
process (clock, reset, enb)
begin
if (signed(din1) < signed(din2)) then
doutMin <= din1;
else
doutMin <= din2;
end if;

if (signed(din1) > signed(din2)) then
doutMax <= din1;
else
doutMax <= din2;
end if;

end process;
end st;
```

7. Write a new VHDL or Verilog testbench file to test the new resource **MINMAX**.

```
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects/8:$vim
testbench.v
```

Listing 2: "testbench.v"

```verilog
`timescale 1ns / 1ps

module testb;

// Inputs
reg [31:0] din1;
reg [31:0] din2;
reg clock;
reg reset;
reg enb;

// Outputs
wire [31:0] doutMin;
wire [31:0] doutMax;

// Instantiate the Unit Under Test (UUT)
minmax uut (
.clock(clock),
.reset(reset),
.enb(enb),
.din1(din1),
.din2(din2),
.doutMin(doutMin),
.doutMax(doutMax)
);

initial begin
// Initialize Inputs
reset = 1;
clock = 1;
enb = 1;
din1 = 23;
din2 = 45;

// Wait 100 ns for global reset to finish
#1000;
// Initialize Inputs
din1 = 123;
din2 = 45;
// Add stimulus here

end
```

```
always
begin
clock = 1'b1;
#20; // high for 20 * timescale = 20 ns

clock = 1'b0;
#20; // low for 20 * timescale = 20 ns
end

endmodule
```

8. Create a new ModelSim project, add above files, compile them, and simulate the hardware to verify the functionality. Once the functionality is verified you can add it to the ASIPmeister via a FHM file.

```
/home/sajjad/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb/workdb/FHM_work
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$cd
ASIPmeister/share/fhmdb/workdb/FHM_work
```

C. INCORPORATE THE RESOURCE IN ASIPMEISTER:

9. Copy ASIPmeister Software to a new directory for example:

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$ cp −rf /AM/ASIPmeister .
```

10. Create FHM file of the resource from "minmax.vhd" using the steps in the Laboratory Script Section 4.4.

11. Copy this FHM file into "*ASIPmeister/share/fhmdb/workdb/FHM_work*"

```
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects/8:$cd
ASIPmeister/share/fhmdb/workdb/FHM_work
sajjad@i83pc57:~/SS21/ ASIPMeisterProjects/8/
ASIPmeister/share/fhmdb/workdb/FHM_work:$vim minmax.fhm
```

Listing 3: "minmax.fhm"

```
<?xml version="1.0" encoding="Shift_JIS" ?>
<FHM>
<model_name> minmax </model_name>

<model>
<design_level> behavior </design_level>
<version> 1.0 </version>
<author> <![CDATA[ Joe Random Hacker ]]> </author>
<affiliation> <![CDATA[ Uni Karlsruhe ]]> </affiliation>
<model_info> <![CDATA[ − ]]> </model_info>

<parameter>
<parameter_value key="bit_width">
<value> 4 </value>
<value> 8 </value>
<value> 16 </value>
<value> 32 </value>
</parameter_value>
</parameter>

<function_description>
<script>
 <![CDATA[
#!/usr/bin/perl
# This script generates register function definition in behavior level
# parameter : bit_width

    if ($#ARGV != 0) {
            print "number_of_parameters_is_wrong.\n";
            print "usage_:_this_script_bit_width\n";
            exit (100);
```

```perl
        }

        $bit_width    = $ARGV[0];
        $msb = $bit_width - 1;

        print <<FHM_DL_FOO;
/** minmax */
function minmax {
        input {
                bit [31:0] din1;
                bit [31:0] din2;
        }
        output {
                bit [31:0] doutMin;
                bit [31:0] doutMax;
        }
        control {
                in clock;
                in reset;
                in enb;
        }
        protocol {
                [enb == '1'] {
                        valid dout;
                }
        }
}
FHM_DL_FOO

        exit (0);
]]>
</script>
</function_description>

<function_conv>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register function definition in behavior level
# parameter : bit_width

if ($#ARGV != 0) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_bit_width\n";
        exit (100);
}

$bit_width    = $ARGV[0];
$msb = $bit_width - 1;

print <<FHM_DL_FOO;
/** minmax */
function minmax {
        input {
                bit [31:0] din1;
                bit [31:0] din2;

        }
        output {
                bit [31:0] doutMin;
                bit [31:0] doutMax;
        }
        control {
                in bit clock;
                in bit reset;
                in bit enb;
        }
        protocol {
                single_cycle_protocol {
                        enb = '1';
                }
        }
}
FHM_DL_FOO

        exit (0);
```

```perl
]]>
</script>
</function_conv>

<function_port>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register port information in behavior level
# parameter : bit_width

if ($#ARGV != 0) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_bit_width\n";
        exit (100);
}

$bit_width      = $ARGV[0];

$msb = $bit_width -1;

print <<FHM_DL_PORTS;
clock in bit ctrl
reset in bit ctrl
enb in bit ctrl
din1 in bit_vector 31 0 data
din2 in bit_vector 31 0 data
doutMin out bit_vector 31 0 data
doutMax out bit_vector 31 0 data
FHM_DL_PORTS

exit (0);
]]>
</script>
</function_port>

<design>
<design_lang> vhdl </design_lang>

<instance>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register instance in behavior level
# parameter : instance_name bit_width

if ($#ARGV != 1) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_instance_name_bit_width\n";
        exit (100);
}

$instance_name = $ARGV[0];
$bit_width     = $ARGV[1];


$msb = $bit_width - 1;

$signals = <<END_SIGNALS;
-- Your signal declaration here
END_SIGNALS

$vhdl = <<END_VHDL;
-- Your vhdl code here
process (clock, reset, enb)
begin
if (signed(din1) < signed(din2)) then
doutMin <= din1;
else
doutMin <= din2;
end if;

if (signed(din1) > signed(din2)) then
doutMax <= din1;
else
doutMax <= din2;
```

```
        end if;
    end process;
END_VHDL


{
        print <<FHM_DL_COMMENTS;
        FHM_DL_COMMENTS
}



print <<FHM_DL_TOP_2;
--    int_port : internal port
--    ext_port : external port

-- Comment :

library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_arith.all;

entity $instance_name is
port (
FHM_DL_TOP_2

print <<FHM_DL_PORTS;
clock    : in std_logic;
reset    : in std_logic;
enb      : in std_logic;
din1 : in  STD_LOGIC_VECTOR (31 downto 0);
din2     : in  STD_LOGIC_VECTOR (31 downto 0);
doutMin    : out  STD_LOGIC_VECTOR (31 downto 0);
doutMax    : out STD_LOGIC_VECTOR (31 downto 0)
);
FHM_DL_PORTS

{
        print <<FHM_DL_ARCH;
        end $instance_name;

        architecture st of $instance_name is
        $signals
        begin
        $vhdl
        end st;

        FHM_DL_ARCH
}

exit (0);
]]>
</script>
</instance>

<entity>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register instance in behavior level
# parameter : instance_name bit_width

if ($#ARGV != 1) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_instance_name_bit_width\n";
        exit (100);
}

$instance_name = $ARGV[0];
$bit_width     = $ARGV[1];


$msb = $bit_width - 1;

{
        print <<FHM_DL_TOP;
```

```
                entity $instance_name is
                port (
                FHM_DL_TOP

        }

        print <<FHM_DL_PORTS;
        clock     : in std_logic;
        reset     : in std_logic;
        enb       : in std_logic;
        din1 : in  STD_LOGIC_VECTOR (31 downto 0);
        din2    : in  STD_LOGIC_VECTOR (31 downto 0);
        doutMin    : out  STD_LOGIC_VECTOR (31 downto 0);
        doutMax    : out STD_LOGIC_VECTOR (31 downto 0)
        );
        FHM_DL_PORTS

        {
                print <<FHM_DL_BOTTOM;
                end $instance_name;
                FHM_DL_BOTTOM
        }
        exit (0);
        ]]>
        </script>
        </entity>

        <testvector>
        <testvector_script>
         <![CDATA[ ]]>
        </testvector_script>
        </testvector>

        <synthesis>
        <parameter></parameter>
        <synthesis_script>
        <script>
         <![CDATA[
        #!/usr/bin/perl
        # This script generates register synthesis script in behavior level
        # parameter : instance_name priority bit_width

        if ($#ARGV != 2) {
                print "number_of_parameters_is_wrong.\n";
                print "usage_:_this_script_instance_name_priority_bit_width\n";
                exit (100);
        }

        $instance_name = $ARGV[0];
        $priority      = $ARGV[1];
        $bit_width     = $ARGV[2];


        if ($priority eq "area"){
                $priority_const = "set_max_area_0";
        }
        elsif ($priority eq "performance"){
                $priority_const = "set_max_delay_-from_all_inputs()_-to_all_outputs()_0";
        }
        elsif ($priority eq "power"){
                $priority_const = "";
        }
        elsif ($priority eq "none"){
                $priority_const = "";
        }
        else {
                print "priority_$priority_is_not_supported.\n";
                exit(100);
        }

        {
                print <<FHM_DL_END_OF_SCRIPT;
                hdlin_auto_save_templates = TRUE

                analyze -f vhdl $instance_name.vhd
```

```
              elaborate $instance_name
              uniquify

              $priority_const

              create_clock −period 10 −waveform{0 5} clock

              compile

              write −hierarchy −output $instance_name.db

              report_area
              report_timing

              quit
              FHM_DL_END_OF_SCRIPT
     }
     exit(0);
     ]]>
     </script>
     </synthesis_script>
     </synthesis>
     </design>

     <estimation>
     <estimation_data>
     <library name="OSAKA">

     <est_type name="shape">
     <est_index name="area">
     <unit> mm2 </unit>
     <translate>
     <translate_value key="gate"> 4201.68 </translate_value>
     <translate_value key="mm2">  1 </translate_value>
     </translate>

     <parameters name="">
     <max>
     <data bit_width="4"> 0.1 </data>
     <data bit_width="8"> 0.1 </data>
     <data bit_width="16"> 0.1 </data>
     <data bit_width="32"> 0.1 </data>
     </max>
     <min>
     <data bit_width="4"> 0.1 </data>
     <data bit_width="8"> 0.1 </data>
     <data bit_width="16"> 0.1 </data>
     <data bit_width="32"> 0.1 </data>
     </min>
     <typ>
     <priority name="area">
     <data bit_width="4"> 0.001 </data>
     <data bit_width="8"> 0.01 </data>
     <data bit_width="16"> 0.1 </data>
     <data bit_width="32"> 0.1 </data>
     </priority>
     <priority name="delay">
     <data bit_width="4"> 0.001 </data>
     <data bit_width="8"> 0.01 </data>
     <data bit_width="16"> 0.1 </data>
     <data bit_width="32"> 0.1 </data>
     </priority>
     <priority name="power">
     <data bit_width="4"> 0.001 </data>
     <data bit_width="8"> 0.01 </data>
     <data bit_width="16"> 0.1 </data>
     <data bit_width="32"> 0.1 </data>
     </priority>
     </typ>
     </parameters>

     </est_index>

     <est_index name="aspect_ratio">
     <!-- Dummy yet -->
```

```xml
        </est_index>

        <est_index name="height">
        <!-- Dummy yet -->
        </est_index>

        <est_index name="width">
        <!-- Dummy yet -->
        </est_index>
        </est_type>

        <est_type name="timing">
        <est_index name="delay">
        <unit> ns </unit>

        <parameters name="">
        <max>
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </max>
        <min>
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </min>
        <typ>
        <priority name="area">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        <priority name="delay">
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </priority>
        <priority name="power">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        </typ>
        </parameters>

        </est_index>

        <est_index name="delay_fullpath">
        <!-- Dummy yet -->
        </est_index>
        </est_type>

        <est_type name="power">
        <est_index name="static_power">
        <unit> mW </unit>
        <parameters name="">
        <max>
        <data bit_width="4"> 2.2203 </data>
        <data bit_width="8"> 4.4270 </data>
        <data bit_width="16"> 8.7214 </data>
        <data bit_width="32"> 17.2327 </data>
        </max>
        <min>
        <data bit_width="4"> 2.2153 </data>
        <data bit_width="8"> 4.3512 </data>
        <data bit_width="16"> 8.5400 </data>
        <data bit_width="32"> 17.0462 </data>
        </min>
        <typ>
        <priority name="area">
        <data bit_width="4"> 2.2159 </data>
```

```xml
<data bit_width="8"> 4.4179 </data>
<data bit_width="16"> 8.7033 </data>
<data bit_width="32"> 17.2202 </data>
</priority>
<priority name="delay">
<data bit_width="4"> 2.2203 </data>
<data bit_width="8"> 4.4270 </data>
<data bit_width="16"> 8.7214 </data>
<data bit_width="32"> 17.2327 </data>
</priority>
<priority name="power">
<data bit_width="4"> 2.2153 </data>
<data bit_width="8"> 4.3512 </data>
<data bit_width="16"> 8.5400 </data>
<data bit_width="32"> 17.0462 </data>
</priority>
</typ>
</parameters>

</est_index>
</est_type>

<est_type name="function_cycle">
<!-- Dummy yet -->
</est_type>

<est_type name="function_power">
<!-- Dummy yet -->
</est_type>
</library>
</estimation_data>

<estimation_method>

<est_type name="shape">

<est_index name="area">
<parameters name="">
<max>
<data bit_width="4"> 0.1 </data>
<data bit_width="8"> 0.1 </data>
<data bit_width="16"> 0.1 </data>
<data bit_width="32"> 0.1 </data>
</max>
<min>
<data bit_width="4"> 0.1 </data>
<data bit_width="8"> 0.1 </data>
<data bit_width="16"> 0.1 </data>
<data bit_width="32"> 0.1 </data>
</min>
<typ>
<priority name="area">
<data bit_width="4"> 0.001 </data>
<data bit_width="8"> 0.01 </data>
<data bit_width="16"> 0.1 </data>
<data bit_width="32"> 0.1 </data>
</priority>
<priority name="delay">
<data bit_width="4"> 0.001 </data>
<data bit_width="8"> 0.01 </data>
<data bit_width="16"> 0.1 </data>
<data bit_width="32"> 0.1 </data>
</priority>
<priority name="power">
<data bit_width="4"> 0.001 </data>
<data bit_width="8"> 0.01 </data>
<data bit_width="16"> 0.1 </data>
<data bit_width="32"> 0.1 </data>
</priority>
</typ>
</parameters>

</est_index>

<est_index name="aspect_ratio">
```

```xml
        <!-- Dummy yet -->

        </est_index>

        <est_index name="height">

        <!-- Dummy yet -->

        </est_index>

        <est_index name="width">

        <!-- Dummy yet -->

        </est_index>

        </est_type>

        <est_type name="timing">

        <est_index name="delay">
        <parameters name="">
        <max>
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </max>
        <min>
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </min>
        <typ>
        <priority name="area">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        <priority name="delay">
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </priority>
        <priority name="power">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        </typ>
        </parameters>


        </est_index>

        <est_index name="delay_fullpath">

        <!-- Dummy yet -->

        </est_index>

        </est_type>

        <est_type name="power">

        <est_index name="static_power">

        <parameters name="">
        <max>
        <data bit_width="4"> 2.2203 </data>
        <data bit_width="8"> 4.4270 </data>
```

```xml
            <data bit_width="16"> 8.7214 </data>
            <data bit_width="32"> 17.2327 </data>
            </max>
            <min>
            <data bit_width="4"> 2.2153 </data>
            <data bit_width="8"> 4.3512 </data>
            <data bit_width="16"> 8.5400 </data>
            <data bit_width="32"> 17.0462 </data>
            </min>
            <typ>
            <priority name="area">
            <data bit_width="4"> 2.2159 </data>
            <data bit_width="8"> 4.4179 </data>
            <data bit_width="16"> 8.7033 </data>
            <data bit_width="32"> 17.2202 </data>
            </priority>
            <priority name="delay">
            <data bit_width="4"> 2.2203 </data>
            <data bit_width="8"> 4.4270 </data>
            <data bit_width="16"> 8.7214 </data>
            <data bit_width="32"> 17.2327 </data>
            </priority>
            <priority name="power">
            <data bit_width="4"> 2.2153 </data>
            <data bit_width="8"> 4.3512 </data>
            <data bit_width="16"> 8.5400 </data>
            <data bit_width="32"> 17.0462 </data>
            </priority>
            </typ>
            </parameters>


            </est_index>

            </est_type>

            <est_type name="function_cycle">

            </est_type>

            <est_type name="function_power">

            </est_type>


            </estimation_method>
            </estimation>

            </model>

            <model>
            <design_level> synthesis </design_level>
            <version> 1.0 </version>
            <author> <![CDATA[ Joe Random Hacker ]]> </author>
            <affiliation> <![CDATA[ Uni Karlsruhe ]]> </affiliation>
            <model_info> <![CDATA[ - ]]> </model_info>

            <parameter>
            <parameter_value key="bit_width">
            <value> 4 </value>
            <value> 8 </value>
            <value> 16 </value>
            <value> 32 </value>
            </parameter_value>
            </parameter>

            <function_description>
            <script>
            <![CDATA[
            #!/usr/bin/perl
            # This script generates register function definition in behavior level
            # parameter : bit_width

            if ($#ARGV != 0) {
                    print "number_of_parameters_is_wrong.\n";
                    print "usage_:_this_script_bit_width\n";
```

```perl
                exit (100);
        }

        $bit_width     = $ARGV[0];
        $msb = $bit_width - 1;

        print <<FHM_DL_FOO;
/** minmax */
function minmax {
        input {
                bit [31:0] din1;
                bit [31:0] din2;
        }
        output {
                bit [31:0] doutMin;
                bit [31:0] doutMax;

        }
        control {
                in clock;
                in reset;
                in enb;
        }
        protocol {
                [enb == '1'] {
                        valid dout;
                }
        }
}
FHM_DL_FOO

        exit (0);
]]>
</script>
</function_description>

<function_conv>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register function definition in behavior level
# parameter : bit_width

if ($#ARGV != 0) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_bit_width\n";
        exit (100);
}

$bit_width     = $ARGV[0];
$msb = $bit_width - 1;

print <<FHM_DL_FOO;
/** minmax */
function minmax {
        input {
                bit [31:0] din1;
                bit [31:0] din2;

        }
        output {
                bit [31:0] doutMin;
                bit [31:0] doutMax;
        }
        control {
                in bit clock;
                in bit reset;
                in bit enb;
        }
        protocol {
                single_cycle_protocol {
                        enb = '1';
                }
        }
}
FHM_DL_FOO
```

```perl
        exit (0);
]]>
</script>
</function_conv>

<function_port>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register port information in behavior level
# parameter : bit_width

if ($#ARGV != 0) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_bit_width\n";
        exit (100);
}

$bit_width     = $ARGV[0];

$msb = $bit_width -1;

print <<FHM_DL_PORTS;
clock in bit ctrl
reset in bit ctrl
enb in bit ctrl
din1 in bit_vector 31 0 data
din2 in bit_vector 31 0 data
doutMin out bit_vector 31 0 data
doutMax out bit_vector 31 0 data
FHM_DL_PORTS

exit (0);
]]>
</script>
</function_port>

<design>
<design_lang> vhdl </design_lang>

<instance>
<script>
<![CDATA[
#!/usr/bin/perl
# This script generates register instance in behavior level
# parameter : instance_name bit_width

if ($#ARGV != 1) {
        print "number_of_parameters_is_wrong.\n";
        print "usage_:_this_script_instance_name_bit_width\n";
        exit (100);
}

$instance_name = $ARGV[0];
$bit_width     = $ARGV[1];


$msb = $bit_width - 1;

$signals = <<END_SIGNALS;
-- Your signal declaration here
END_SIGNALS

$vhdl = <<END_VHDL;
-- Your vhdl code here
process (clock, reset, enb)
begin
if (signed(din1) < signed(din2)) then
doutMin <= din1;
else
doutMin <= din2;
end if;

if (signed(din1) > signed(din2)) then
doutMax <= din1;
```

```
            else
            doutMax <= din2;
            end if;
            end process;
            END_VHDL


            {
                    print <<FHM_DL_COMMENTS;
                    FHM_DL_COMMENTS
            }



            print <<FHM_DL_TOP_2;
            --    int_port : internal port
            --    ext_port : external port

            -- Comment :

            library IEEE;
            use IEEE.std_logic_1164.all;
            use IEEE.std_logic_arith.all;

            entity $instance_name is
            port (
            FHM_DL_TOP_2

            print <<FHM_DL_PORTS;
            clock    : in std_logic;
            reset    : in std_logic;
            enb      : in std_logic;
            din1 : in  STD_LOGIC_VECTOR (31 downto 0);
            din2    : in  STD_LOGIC_VECTOR (31 downto 0);
            doutMin   : out  STD_LOGIC_VECTOR (31 downto 0);
            doutMax   : out STD_LOGIC_VECTOR (31 downto 0)
            );
            FHM_DL_PORTS

            {
                    print <<FHM_DL_ARCH;
                    end $instance_name;

                    architecture st of $instance_name is
                    $signals
                    begin
                    $vhdl
                    end st;

                    FHM_DL_ARCH
            }

            exit (0);
            ]]>
            </script>
            </instance>

            <entity>
            <script>
            <![CDATA[
            #!/usr/bin/perl
            # This script generates register instance in behavior level
            # parameter : instance_name bit_width

            if ($#ARGV != 1) {
                    print "number_of_parameters_is_wrong.\n";
                    print "usage_:_this_script_instance_name_bit_width\n";
                    exit (100);
            }

            $instance_name = $ARGV[0];
            $bit_width     = $ARGV[1];


            $msb = $bit_width - 1;
```

```perl
        {
                print <<FHM_DL_TOP;

                entity $instance_name is
                port (
                FHM_DL_TOP

        }

        print <<FHM_DL_PORTS;
        clock     : in std_logic;
        reset     : in std_logic;
        enb       : in std_logic;
        din1 : in   STD_LOGIC_VECTOR (31 downto 0);
        din2    : in   STD_LOGIC_VECTOR (31 downto 0);
        doutMin    : out   STD_LOGIC_VECTOR (31 downto 0);
        doutMax    : out STD_LOGIC_VECTOR (31 downto 0)
        );
        FHM_DL_PORTS

        {
                print <<FHM_DL_BOTTOM;
                end $instance_name;
                FHM_DL_BOTTOM
        }
        exit (0);
        ]]>
        </script>
        </entity>

        <testvector>
        <testvector_script>
         <![CDATA[ ]]>
        </testvector_script>
        </testvector>

        <synthesis>
        <parameter></parameter>
        <synthesis_script>
        <script>
         <![CDATA[
        #!/usr/bin/perl
        # This script generates register synthesis script in behavior level
        # parameter : instance_name priority bit_width

        if ($#ARGV != 2) {
                print "number_of_parameters_is_wrong.\n";
                print "usage_:_this_script_instance_name_priority_bit_width\n";
                exit (100);
        }

        $instance_name = $ARGV[0];
        $priority       = $ARGV[1];
        $bit_width      = $ARGV[2];


        if ($priority eq "area"){
                $priority_const = "set_max_area_0";
        }
        elsif ($priority eq "performance"){
                $priority_const = "set_max_delay_-from_all_inputs()_-to_all_outputs()_0";
        }
        elsif ($priority eq "power"){
                $priority_const = "";
        }
        elsif ($priority eq "none"){
                $priority_const = "";
        }
        else {
                print "priority_$priority_is_not_supported.\n";
                exit(100);
        }

        {
                print <<FHM_DL_END_OF_SCRIPT;
                hdlin_auto_save_templates = TRUE
```

```
                  analyze -f vhdl $instance_name.vhd

                  elaborate $instance_name
                  uniquify

                  $priority_const

                  create_clock -period 10 -waveform{0 5} clock

                  compile

                  write -hierarchy -output $instance_name.db

                  report_area
                  report_timing

                  quit
                  FHM_DL_END_OF_SCRIPT
        }
        exit(0);
        ]]>
        </script>
        </synthesis_script>
        </synthesis>
        </design>

        <estimation>
        <estimation_data>
        <library name="OSAKA">

        <est_type name="shape">
        <est_index name="area">
        <unit> mm2 </unit>
        <translate>
        <translate_value key="gate"> 4201.68 </translate_value>
        <translate_value key="mm2">  1 </translate_value>
        </translate>

        <parameters name="">
        <max>
        <data bit_width="4"> 0.1 </data>
        <data bit_width="8"> 0.1 </data>
        <data bit_width="16"> 0.1 </data>
        <data bit_width="32"> 0.1 </data>
        </max>
        <min>
        <data bit_width="4"> 0.1 </data>
        <data bit_width="8"> 0.1 </data>
        <data bit_width="16"> 0.1 </data>
        <data bit_width="32"> 0.1 </data>
        </min>
        <typ>
        <priority name="area">
        <data bit_width="4"> 0.001 </data>
        <data bit_width="8"> 0.01 </data>
        <data bit_width="16"> 0.1 </data>
        <data bit_width="32"> 0.1 </data>
        </priority>
        <priority name="delay">
        <data bit_width="4"> 0.001 </data>
        <data bit_width="8"> 0.01 </data>
        <data bit_width="16"> 0.1 </data>
        <data bit_width="32"> 0.1 </data>
        </priority>
        <priority name="power">
        <data bit_width="4"> 0.001 </data>
        <data bit_width="8"> 0.01 </data>
        <data bit_width="16"> 0.1 </data>
        <data bit_width="32"> 0.1 </data>
        </priority>
        </typ>
        </parameters>

        </est_index>
```

```xml
        <est_index name="aspect_ratio">
        <!-- Dummy yet -->
        </est_index>

        <est_index name="height">
        <!-- Dummy yet -->
        </est_index>

        <est_index name="width">
        <!-- Dummy yet -->
        </est_index>
        </est_type>

        <est_type name="timing">
        <est_index name="delay">
        <unit> ns </unit>

        <parameters name="">
        <max>
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </max>
        <min>
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </min>
        <typ>
        <priority name="area">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        <priority name="delay">
        <data bit_width="4"> 0.72 </data>
        <data bit_width="8"> 0.72 </data>
        <data bit_width="16"> 0.72 </data>
        <data bit_width="32"> 0.72 </data>
        </priority>
        <priority name="power">
        <data bit_width="4"> 0.75 </data>
        <data bit_width="8"> 0.75 </data>
        <data bit_width="16"> 0.75 </data>
        <data bit_width="32"> 0.75 </data>
        </priority>
        </typ>
        </parameters>

        </est_index>

        <est_index name="delay_fullpath">
        <!-- Dummy yet -->
        </est_index>
        </est_type>

        <est_type name="power">
        <est_index name="static_power">
        <unit> mW </unit>
        <parameters name="">
        <max>
        <data bit_width="4"> 2.2203 </data>
        <data bit_width="8"> 4.4270 </data>
        <data bit_width="16"> 8.7214 </data>
        <data bit_width="32"> 17.2327 </data>
        </max>
        <min>
        <data bit_width="4"> 2.2153 </data>
        <data bit_width="8"> 4.3512 </data>
        <data bit_width="16"> 8.5400 </data>
        <data bit_width="32"> 17.0462 </data>
        </min>
        <typ>
```

```xml
                    <priority name="area">
                    <data bit_width="4"> 2.2159 </data>
                    <data bit_width="8"> 4.4179 </data>
                    <data bit_width="16"> 8.7033 </data>
                    <data bit_width="32"> 17.2202 </data>
                    </priority>
                    <priority name="delay">
                    <data bit_width="4"> 2.2203 </data>
                    <data bit_width="8"> 4.4270 </data>
                    <data bit_width="16"> 8.7214 </data>
                    <data bit_width="32"> 17.2327 </data>
                    </priority>
                    <priority name="power">
                    <data bit_width="4"> 2.2153 </data>
                    <data bit_width="8"> 4.3512 </data>
                    <data bit_width="16"> 8.5400 </data>
                    <data bit_width="32"> 17.0462 </data>
                    </priority>
                    </typ>
                    </parameters>

                    </est_index>
                    </est_type>

                    <est_type name="function_cycle">
                    <!-- Dummy yet -->
                    </est_type>

                    <est_type name="function_power">
                    <!-- Dummy yet -->
                    </est_type>
                    </library>
                    </estimation_data>

                    <estimation_method>

                    <est_type name="shape">

                    <est_index name="area">
                    <parameters name="">
                    <max>
                    <data bit_width="4"> 0.1 </data>
                    <data bit_width="8"> 0.1 </data>
                    <data bit_width="16"> 0.1 </data>
                    <data bit_width="32"> 0.1 </data>
                    </max>
                    <min>
                    <data bit_width="4"> 0.1 </data>
                    <data bit_width="8"> 0.1 </data>
                    <data bit_width="16"> 0.1 </data>
                    <data bit_width="32"> 0.1 </data>
                    </min>
                    <typ>
                    <priority name="area">
                    <data bit_width="4"> 0.001 </data>
                    <data bit_width="8"> 0.01 </data>
                    <data bit_width="16"> 0.1 </data>
                    <data bit_width="32"> 0.1 </data>
                    </priority>
                    <priority name="delay">
                    <data bit_width="4"> 0.001 </data>
                    <data bit_width="8"> 0.01 </data>
                    <data bit_width="16"> 0.1 </data>
                    <data bit_width="32"> 0.1 </data>
                    </priority>
                    <priority name="power">
                    <data bit_width="4"> 0.001 </data>
                    <data bit_width="8"> 0.01 </data>
                    <data bit_width="16"> 0.1 </data>
                    <data bit_width="32"> 0.1 </data>
                    </priority>
                    </typ>
                    </parameters>

                    </est_index>
```

```xml
<est_index name="aspect_ratio">

<!-- Dummy yet -->

</est_index>

<est_index name="height">

<!-- Dummy yet -->

</est_index>

<est_index name="width">

<!-- Dummy yet -->

</est_index>

</est_type>

<est_type name="timing">

<est_index name="delay">
<parameters name="">
<max>
<data bit_width="4"> 0.75 </data>
<data bit_width="8"> 0.75 </data>
<data bit_width="16"> 0.75 </data>
<data bit_width="32"> 0.75 </data>
</max>
<min>
<data bit_width="4"> 0.72 </data>
<data bit_width="8"> 0.72 </data>
<data bit_width="16"> 0.72 </data>
<data bit_width="32"> 0.72 </data>
</min>
<typ>
<priority name="area">
<data bit_width="4"> 0.75 </data>
<data bit_width="8"> 0.75 </data>
<data bit_width="16"> 0.75 </data>
<data bit_width="32"> 0.75 </data>
</priority>
<priority name="delay">
<data bit_width="4"> 0.72 </data>
<data bit_width="8"> 0.72 </data>
<data bit_width="16"> 0.72 </data>
<data bit_width="32"> 0.72 </data>
</priority>
<priority name="power">
<data bit_width="4"> 0.75 </data>
<data bit_width="8"> 0.75 </data>
<data bit_width="16"> 0.75 </data>
<data bit_width="32"> 0.75 </data>
</priority>
</typ>
</parameters>


</est_index>

<est_index name="delay_fullpath">

<!-- Dummy yet -->

</est_index>

</est_type>

<est_type name="power">

<est_index name="static_power">

<parameters name="">
<max>
```

```
            <data bit_width="4"> 2.2203 </data>
            <data bit_width="8"> 4.4270 </data>
            <data bit_width="16"> 8.7214 </data>
            <data bit_width="32"> 17.2327 </data>
            </max>
            <min>
            <data bit_width="4"> 2.2153 </data>
            <data bit_width="8"> 4.3512 </data>
            <data bit_width="16"> 8.5400 </data>
            <data bit_width="32"> 17.0462 </data>
            </min>
            <typ>
            <priority name="area">
            <data bit_width="4"> 2.2159 </data>
            <data bit_width="8"> 4.4179 </data>
            <data bit_width="16"> 8.7033 </data>
            <data bit_width="32"> 17.2202 </data>
            </priority>
            <priority name="delay">
            <data bit_width="4"> 2.2203 </data>
            <data bit_width="8"> 4.4270 </data>
            <data bit_width="16"> 8.7214 </data>
            <data bit_width="32"> 17.2327 </data>
            </priority>
            <priority name="power">
            <data bit_width="4"> 2.2153 </data>
            <data bit_width="8"> 4.3512 </data>
            <data bit_width="16"> 8.5400 </data>
            <data bit_width="32"> 17.0462 </data>
            </priority>
            </typ>
            </parameters>


            </est_index>

            </est_type>

            <est_type name="function_cycle">

            </est_type>

            <est_type name="function_power">

            </est_type>


            </estimation_method>
            </estimation>

            </model>

            </FHM>
```

12. Add the new hardware into the ASIPmeister resource list by editing "***ASIPmeister/share/fh-mdb/fhmdbstruct***" and add the line "***<model>minmax</model>***" in the FHM_WORK class.

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb/:$vim fhmdbstruct
```

Listing 4: "fhmdbstruct"

```
<FHM_Structure>
<library name="basicfhmdb">
<class name="computational">
<model>adder</model>
<model>adder0</model>
<model>alu</model>
<model>alu0</model>
<model>barrelshifter0</model>
<model>divider</model>
<model>divider0</model>
<model>extender</model>
```

```
<model>extender0</model>
<model>mini_alu</model>
<model>multiplexor</model>
<model>multiplexor0</model>
<model>multiplier</model>
<model>multiplier0</model>
<model>rotator</model>
<model>shifter</model>
<model>shifter0</model>
</class>
<class name="storage">
<model>register</model>
<model>register0</model>
<model>registerfile</model>
<model>registerfile0</model>
</class>
</library>
<library name="workdb">
<class name="FHM_work">
<model>browregfile</model>
<model>dmau0</model>
<model>dummy_register</model>
<model>fwu</model>
<model>fwu0</model>
<model>genport0</model>
<model>imau0</model>
<model>mifu</model>
<model>pcu</model>
<model>pcu0</model>
<model>sleeper0</model>
<model>wire0</model>
<model>wire_in</model>
<model>wire_inout</model>
<model>wire_out</model>
<model>clamp</model>
<model>stepsize</model>
<model>index</model>
<model>adpcm</model>
<model>adpcmdecode</model>
<model>adpcmdecode2</model>
<model>minmax</model>
</class>
</library>
</FHM_Structure>
```

13. Set the ASIPmeister PATH to this ASIPmeister either in .bashrc or manually by PATH variable each time.

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8/ASIPmeister/share/fhmdb:$cd ../../../
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$ PATH=/home/sajjad/SS21/ASIPMeisterProjects/8/
    ↪ ASIPmeister/bin/:$PATH
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$ which ASIPmeister
~/SS21/ASIPMeisterProjects/8/ASIPmeister/bin/ASIPmeister
```

D. ADD CUSTOM INSTRUCTIONS:

14. We will implement two minmax instruction using existing resources and using new resource.

D.1   USING EXISTING RESOURCES

15. For each ASIPmeister CPU create a separate directory in "**ASIPMeisterProjects**". For example, copy TEMPLATE PROJECT and rename it e.g., "**brownie**" for ASIPmeister CPU "**brow-std32.pdb**"

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$cp -r
/home/ces-asip00/epp/ASIPMeisterProjects/TEMPLATE_PROJECT ./brownie
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8:$ls
brownie
```

16. Set the parameters and settings of the ASIP project in "***env_settings***"

17. Open ASIPMeister CPU in the respective directory i.e., in brownie

```
sajjad@i83pc57:~/SS21/Session1/ASIPMeisterProjects/brownie:$ASIPmeister
browstd32.pdb &
```

18. Modify the CPU in ASIPmeister. Resource Declaration will look like the following. "***MMX***" is the hardware instance of "***minmax.vhd***". FWU2 and FWU3 are Forwarding Units for forwarding 2nd destination operand.
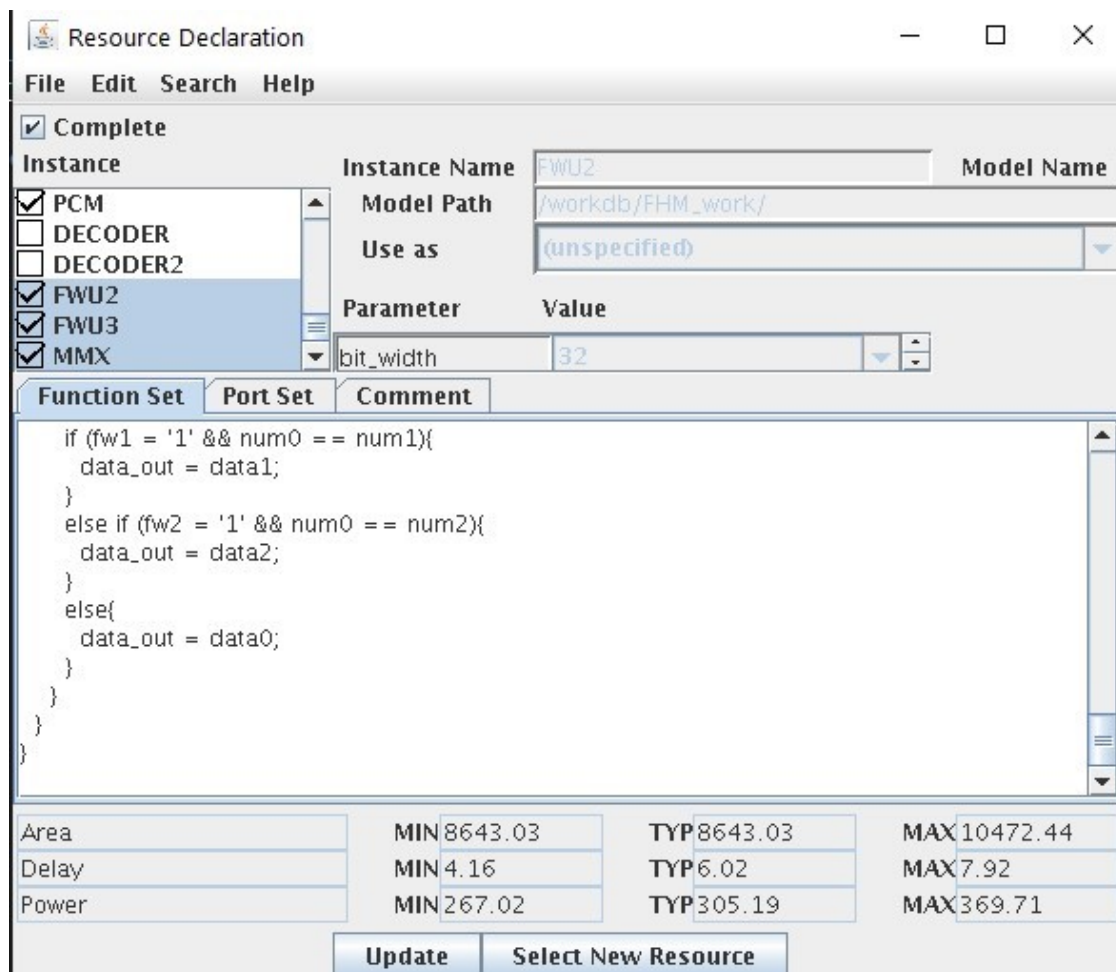

Figure 1

19. Then, create a new instruction formats with two input and two output registers. And create two different instructions which will use existing and new resources respectively.

20. The Micro-Op description for the instruction MMAXS, which will be using the existing resources of ASIPmeister.

D.2   USING NEW RESOURCES

21. The Micro-Op description for the instruction **MMAX**, which will be using the new resource **MMX** of ASIPmeister.
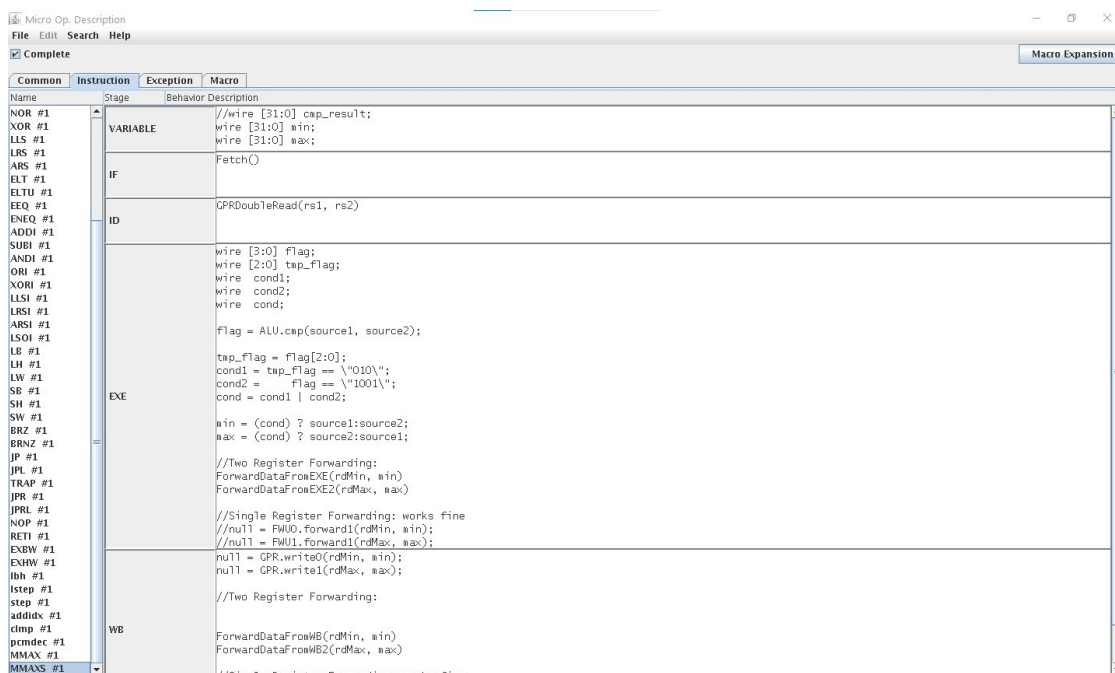
22. C Definition may look like this.
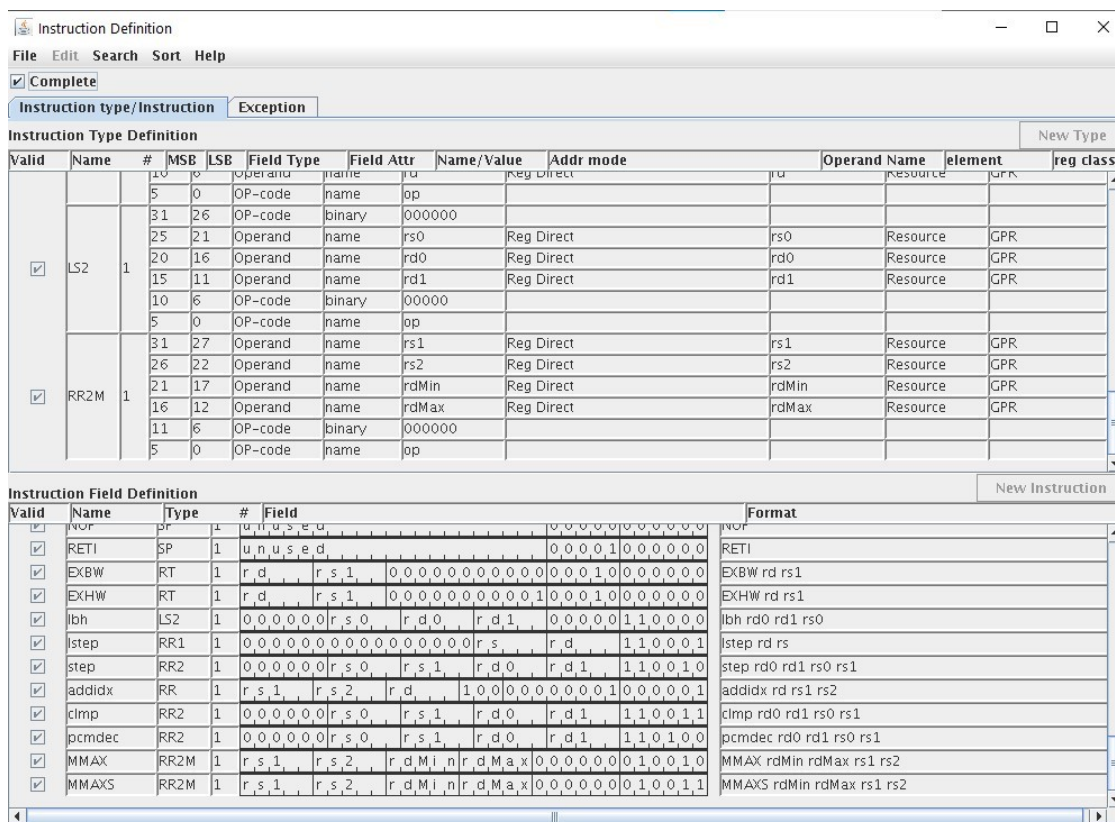
Figure 2

Figure 3

23. Generate the required compiler and VHDL files. A "***meister***" directory will be created in your ASIP project directory i.e. in "***brownie***"

24. Create a ModelSim project in the relevant directory of the current project, compile it and make it ready for the simulation.
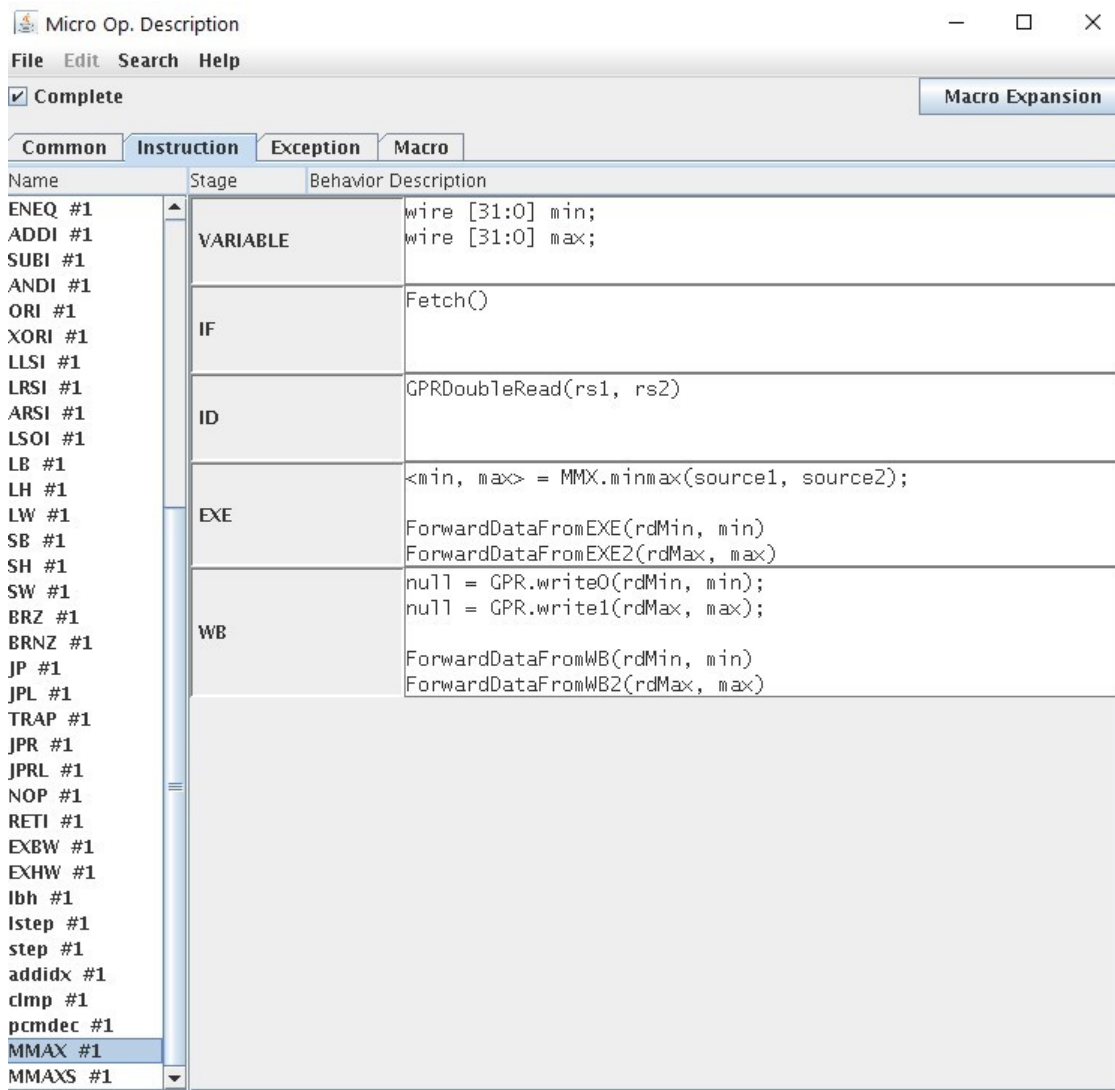
E. USING INSTRUCTIONS:

Figure 4

25. Goto the "Applications" folder and create different applications to test each.

E.1   ORIGINAL APPLICATION

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8/
brownie/Applications/ testMinMaxAPP:$ vim test.c
```

Listing 5: "test.c using original"

```c
#define max(a, b) ( (a)>(b) ? (a):(b) )
#define min(a, b) ( (a)<(b) ? (a):(b))
unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0,0};
int main() {
        A[4]   = max(A[0], A[1]); // Maximum
        A[5]   = min(A[0], A[1]); // Minimum
        return 0;
}
```

E.2   APPLICATION WITH MMAXS (Existing Resources)

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8/
brownie/Applications/ testMinMaxSW:$ vim test.c
```
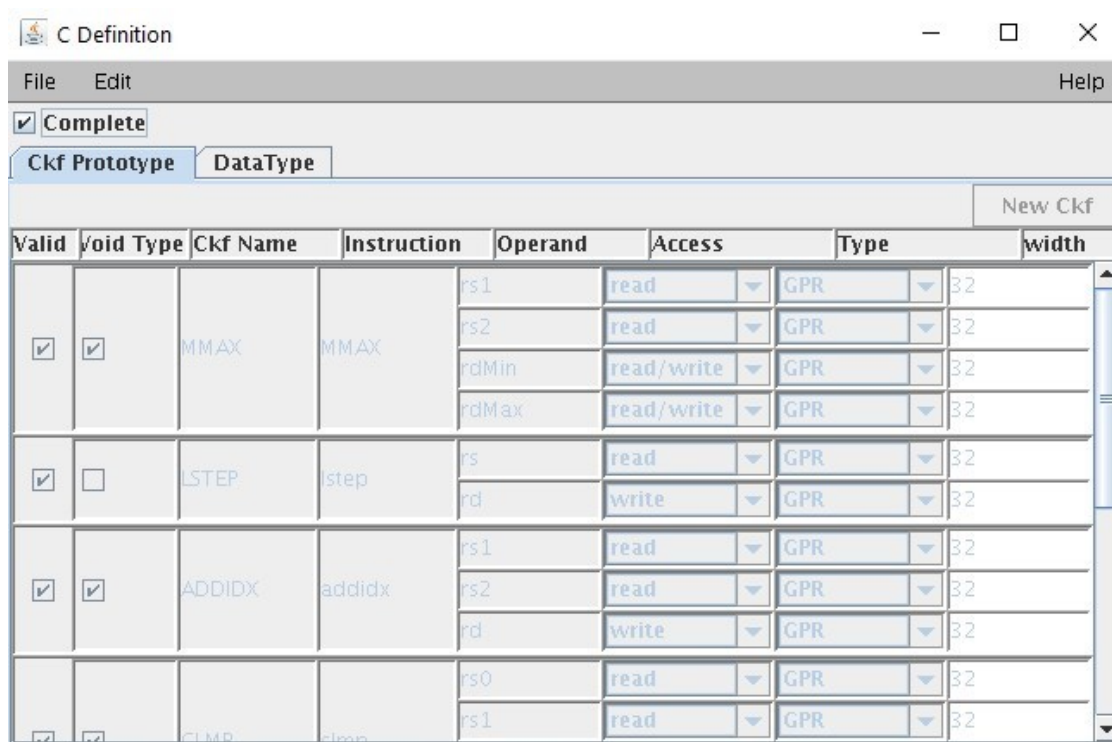
Figure 5

Listing 6: "test.c using existing resources"

```
unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0};
int main() {
        __asm__ volatile (
        "mmaxs %[out1], %[out2], %[op1], %[op2]\n\t"
        : [out1] "=&r" (A[5]), [out2] "=&r" (A[4])
        : [op1] "r" (A[0]), [op2] "r" (A[1])
        );
        return 0;
}
```

E.3   APPLICATION WITH MMAX (New Resources)

```
sajjad@i83pc57:~/SS21/ASIPMeisterProjects/8/
brownie/Applications/ testMinMaxHW:$ vim test.c
```

Listing 7: "test.c using added resources"

```
unsigned int A[10] = { 32, 45, 0,0,0,0,0,0,0};
int main() {
        __asm__ volatile (
        "mmax %[out1], %[out2], %[op1], %[op2]\n\t"
        : [out1] "=&r" (A[5]), [out2] "=&r" (A[4])
        : [op1] "r" (A[0]), [op2] "r" (A[1])
        );

        return 0;
}
```

E.4   MODELSIM SIMULATIONS

26. ModelSim simulations produces following values:

```
testMinMaxAPP
_____
# ** Failure: SUCCESSFUL: Simulation End.
#    Time: 2205 ns  Iteration: 0  Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects
   ↪ /8/dlx_opt_performance_1/ModelSim/tb_browstd32.vhd
```

```
# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx_opt_performance_1/ModelSim
    ↪ /tb_browstd32.vhd line 603

testMinMaxSW
─────────────
# ** Failure: SUCCESSFUL: Simulation End.
#    Time: 1195 ns  Iteration: 0  Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects
    ↪ /8/dlx_opt_performance_1/ModelSim/tb_browstd32.vhd
# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx_opt_performance_1/ModelSim
    ↪ /tb_browstd32.vhd line 603

testMinMaxHW
─────────────
# ** Failure: SUCCESSFUL: Simulation End.
#    Time: 1195 ns  Iteration: 0  Process: /test/dmem File: /home/sajjad/SS21/ASIPMeisterProjects
    ↪ /8/dlx_opt_performance_1/ModelSim/tb_browstd32.vhd
# Break in Process dmem at /home/sajjad/SS21/ASIPMeisterProjects/8/dlx_opt_performance_1/ModelSim
    ↪ /tb_browstd32.vhd line 603
```