EE 789 - Algorithmic Design of Digital Systems

Assignment - 1
Shift and Subtract Division

Name:	**Satvikkumar Patel**
Roll number:	23M1117
Program:	M.tech (EE5 - Electronic Systems)
Department:	Electrical Engineering
Date of Sub:	28th Sep, 2023

Shift and Subtract Divider

Design a shift and subtract (long-division) 8-bit (unsigned) divider in RTL.
(a) Describe the algorithm using Aa, and write a testbench for the design.
(b) Generate the VHDL and verify it using the GHDL simulator and a testbench.

Section: a

Explanation:

The shift_and_subtract_div algorithm is used to perform long division using a shift and subtract method. Here is a step-by-step explanation of the algorithm:

1. Initialize the variables:
- RST_STATE: Represents the reset state of the algorithm.
- LOOP_STATE: Represents the loop state of the algorithm.
- DONE_STATE: Represents the done state of the algorithm.

2. Define the module
- "shift_and_subtract_div" with input variables "a" and "b" of 8-bit unsigned integers and an output variable "quotient" of 8-bit unsigned integers.

3. Inside the module, define the following variables:
- temp: Represents the dividend, which is initialized with the value of "a" concatenated with 7 zeros.
- machine_state: Represents the current state of the algorithm.
- temp_quotient: Represents the current quotient, which is initialized with 8 zeros.
- dividend: Represents the current dividend, which is initialized with the value of "temp".
- counter: Represents the current counter, which is initialized with 8.

4. Enter the loop block:
- Use a merge statement to determine the next values of the variables based on the machine state.
- Use phi statements to update the values of the variables based on the next values.

5. Determine the next machine state:
- Use an excmux statement to select the next state based on the machine state and the counter value.
- If the machine state is RST_STATE, set the next state to LOOP_STATE.
- If the machine state is LOOP_STATE and the counter is 0, set the next state to DONE_STATE.
- If the machine state is DONE_STATE, set the next state to DONE_STATE.

6. Determine the next quotient:
- Use a mux statement to select the next quotient based on the machine state and the comparison between the upper 8 bits of the dividend and the divisor "b".
- If the upper 8 bits of the dividend are less than the divisor "b", set the next quotient to the current quotient concatenated with a zero.
- If the upper 8 bits of the dividend are greater than or equal to the divisor "b", set the next quotient to the current quotient concatenated with a one.

7. Determine the next dividend:
- Use a mux statement to select the next dividend based on the machin state and the comparison between the upper 8 bits of the dividend and the divisor "b".
- If the upper 8 bits of the dividend are less than the divisor "b", set the next dividend to the current dividend shifted right by 1 bit and concatenated with a zero.
- If the upper 8 bits of the dividend are greater than or equal to the divisor "b", set the next dividend to the current dividend minus the divisor "b" and shifted right by 1 bit.

8. Determine the next counter:
- Use a mux statement to select the next counter based on the machine state and the comparison between the counter and zero.
- If the counter is greater than zero, decrement the counter by 1.
- If the counter is zero, set the next counter to zero.

9. Determine the continue_flag:
- Set the continue_flag to true if the counter is less than 9, indicating that the algorithm should continue looping.
-Use an if statement to check the continue_flag and place a loopback label to continue the loop if the flag is true.

This algorithm performs long division by repeatedly subtracting the divisor "b" from the dividend "a" and shifting the dividend right by 1 bit until the counter reaches zero. The quotient is determined by concatenating zeros or ones based on the comparison between the upper 8 bits of the dividend and the divisor.

Test Bench

```

#include <signal.h>
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <pthread.h>
#include <pthreadUtils.h>
#include <Pipes.h>
#include <pipeHandler.h>
#ifndef SW
#include "vhdlCStubs.h"
#endif

int main(int argc, char* argv[])
{
  uint8_t a,b,quotient;

  quotient = shift_and_subtract_div(12,6);
  fprintf(stdout, "shift_and_subtract_div(12,6) = %d\n",quotient);
  return(0);
}
```

Aa algorithm

```
$constant RST_STATE: $uint<2>:=0
$constant LOOP_STATE: $uint<2> :=1
$constant DONE_STATE: $uint<2> :=2


$module [shift_and_subtract_div]
    $in (a b:$uint<8>)
    $out (quotient :$uint<8>)
    $is
{
    temp := ($concat $zero<7> a $zero<1>)
    $branchblock[loop]
    {
        $merge $entry loopback
            $phi current_state := RST_STATE $on $entry next_current_state $on
   loopback
            $phi temp_quotient := $zero<8> $on $entry next_temp_quotient $on
   loopback
            $phi dividend := temp $on $entry next_dividend $on loopback
            $phi counter := 8 $on $entry next_counter $on loopback
      $endmerge

        next_current_state :=
    ($excmux
            (current_state == RST_STATE) LOOP_STATE
            (current_state == LOOP_STATE)
            ($mux (counter == 0) DONE_STATE LOOP_STATE)
            (current_state == DONE_STATE) DONE_STATE)


        next_temp_quotient :=
      ($mux (current_state == LOOP_STATE)
              ($mux (($slice dividend 15 8) < b)
                       ($concat ($slice temp_quotient 6 0) $zero<1>)
                       ($concat ($slice temp_quotient 6 0) $one<1>))
             temp_quotient)

        next_dividend :=
            ($mux (current_state == LOOP_STATE)
                  ($mux (($slice dividend 15 8) < b)
              ($concat ($slice dividend 14 0) $zero<1>  )
              ($concat ($slice (($slice dividend 15 8)-b) 6 0) ($slice dividend
    7 0) $zero<1>))
             dividend)

        next_counter :=
      ($mux (current_state == LOOP_STATE )
         ($mux (counter > 0) (counter-1) $zero<4> )
         ($concat $one<1>  $zero<3>))

        continue_flag := (counter > 0)

        $if continue_flag $then
        $place[loopback]
        $endif

    } (temp_quotient => temp_quotient)
  quotient := temp_quotient
}
```

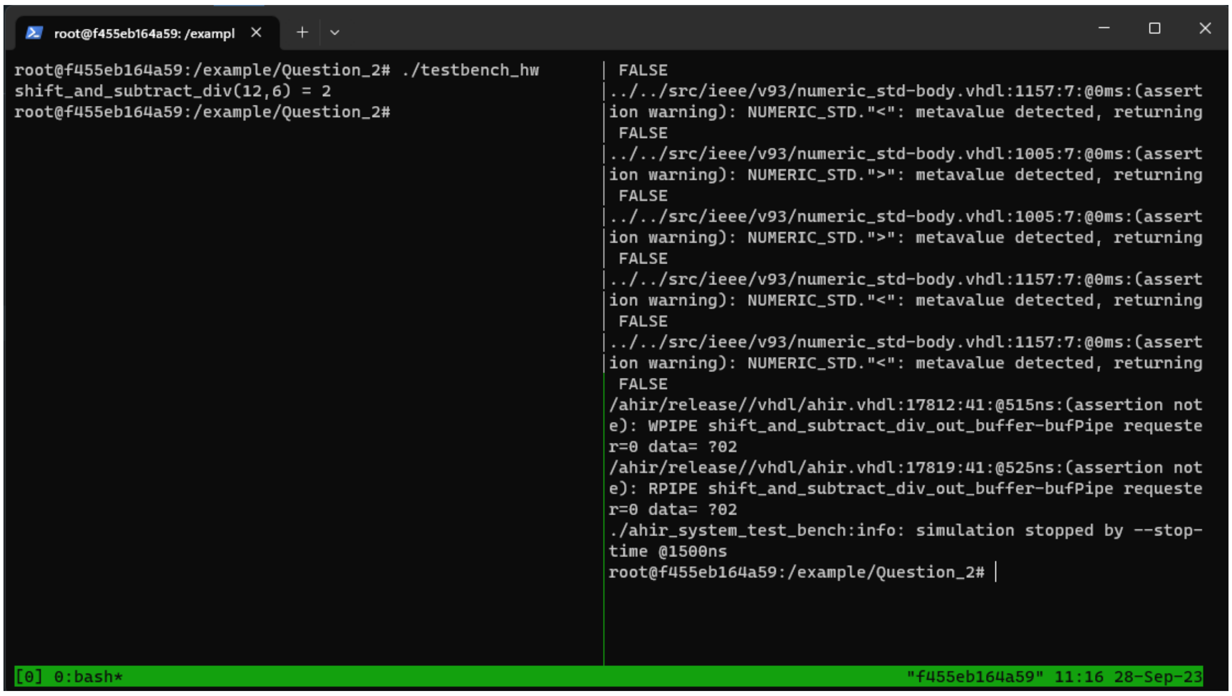
Makefile

```
# build software version of testbench (to check the "desired behaviour")
AHIR_INCLUDE=$(AHIR_RELEASE)/include
AHIR_LIB=$(AHIR_RELEASE)/lib
VHDL_LIB=$(AHIR_RELEASE)/vhdl
VHDL_VHPI_LIB=$(AHIR_RELEASE)/CtestBench/vhdl
FUNCTIONLIB=$(AHIR_RELEASE)/functionLibrary/
SRC=./src
all: HW
TOVC:aalink aa2vc
VC2VHDL: vc2vhdl  vhdlsim
AA2VHDLSIM: aa2vc  vc2vhdl  vhdlsim
TOVHDL:TOVC vc2vhdl


TOPMODULES=-t shift_and_subtract_div



# five steps from C to vhdl simulator.
HW: aalink aa2vc  vc2vhdl  vhdltb ghdlsim

AA2VHDL: aa2vc vc2vhdl vhdlsim

# Aa to vC
aalink: $(SRC)/shift_and_subtract_div.aa
  AaLinkExtMem $(SRC)/shift_and_subtract_div.aa | vcFormat > prog.linked.aa
  AaOpt -B prog.linked.aa | vcFormat > prog.linked.opt.aa

aa2vc: prog.linked.opt.aa
  Aa2VC -O -C prog.linked.opt.aa | vcFormat > prog.vc

# vC to VHDL
vc2vhdl: prog.vc
  vc2vhdl -U -O -v -a -C -e ahir_system -w -s ghdl $(TOPMODULES) -f prog.vc
  vhdlFormat < ahir_system_global_package.unformatted_vhdl >
    ahir_system_global_package.vhdl
  vhdlFormat < ahir_system.unformatted_vhdl > ahir_system.vhdl
  vhdlFormat < ahir_system_test_bench.unformatted_vhdl > ahir_system_test_bench
    .vhdl

# build testbench and ghdl executable
# note the use of libVhpi in building the testbench.
vhdltb: ahir_system.vhdl ahir_system_test_bench.vhdl $(SRC)/testbench.c
    vhdlCStubs.h vhdlCStubs.c
  gcc -c vhdlCStubs.c  -I$(SRC) -I./ -I$(AHIR_INCLUDE)
  gcc -c $(SRC)/testbench.c -I$(AHIR_INCLUDE) -I$(SRC) -I./
  gcc -o testbench_hw testbench.o vhdlCStubs.o  -L$(AHIR_LIB) -
    lSocketLibPipeHandler -lpthread

ghdlsim: ahir_system.vhdl ahir_system_test_bench.vhdl $(SRC)/testbench.c
    vhdlCStubs.h vhdlCStubs.c
  ghdl --clean
  ghdl --remove
  ghdl -i --work=GhdlLink  $(VHDL_LIB)/GhdlLink.vhdl
  ghdl -i --work=aHiR_ieee_proposed  $(VHDL_LIB)/aHiR_ieee_proposed.vhdl
  ghdl -i --work=ahir  $(VHDL_LIB)/ahir.vhdl
  ghdl -i --work=work ahir_system_global_package.vhdl
  ghdl -i --work=work ahir_system.vhdl
  ghdl -i --work=work ahir_system_test_bench.vhdl
  ghdl -m --work=work -Wl,-L$(AHIR_LIB) -Wl,-lVhpi ahir_system_test_bench

clean:
  rm -rf *.o* *.cf *.*vhdl vhdlCStubs.* *.vcd in_data* out_data* testbench_sw testbench_hw ahir_system_test_bench vhpi.log *.aa *.vc *.lso xst *.ngc *_xmsgs *.xrpt pipeHandler.log *.srp *.ghw *.dot

PHONY: all clean
```

Section: b

The test bench is setting $a = (12)_{dec}$ and $b = (6)_{dec}$, and the algorithm generates the output $product = (2)_{dec}$ as expected.

GHDL Simulation

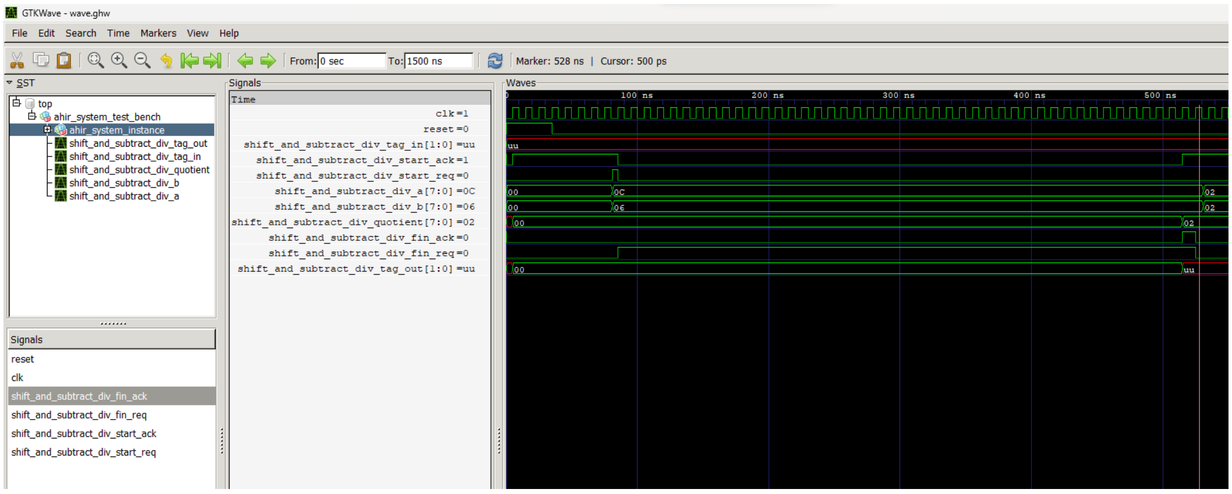


```
root@f455eb164a59:/example/Question_2# ./testbench_hw
shift_and_subtract_div(12,6) = 2
root@f455eb164a59:/example/Question_2#
```

```
 FALSE
../../src/ieee/v93/numeric_std-body.vhdl:1157:7:@0ms:(assert
ion warning): NUMERIC_STD."<": metavalue detected, returning
 FALSE
../../src/ieee/v93/numeric_std-body.vhdl:1005:7:@0ms:(assert
ion warning): NUMERIC_STD.">": metavalue detected, returning
 FALSE
../../src/ieee/v93/numeric_std-body.vhdl:1005:7:@0ms:(assert
ion warning): NUMERIC_STD.">": metavalue detected, returning
 FALSE
../../src/ieee/v93/numeric_std-body.vhdl:1157:7:@0ms:(assert
ion warning): NUMERIC_STD."<": metavalue detected, returning
 FALSE
../../src/ieee/v93/numeric_std-body.vhdl:1157:7:@0ms:(assert
ion warning): NUMERIC_STD."<": metavalue detected, returning
 FALSE
/ahir/release//vhdl/ahir.vhdl:17812:41:@515ns:(assertion not
e): WPIPE shift_and_subtract_div_out_buffer-bufPipe requeste
r=0 data= ?02
/ahir/release//vhdl/ahir.vhdl:17819:41:@525ns:(assertion not
e): RPIPE shift_and_subtract_div_out_buffer-bufPipe requeste
r=0 data= ?02
./ahir_system_test_bench:info: simulation stopped by --stop-
time @1500ns
root@f455eb164a59:/example/Question_2#
```

```
[0] 0:bash*                                    "f455eb164a59" 11:16 28-Sep-23
```

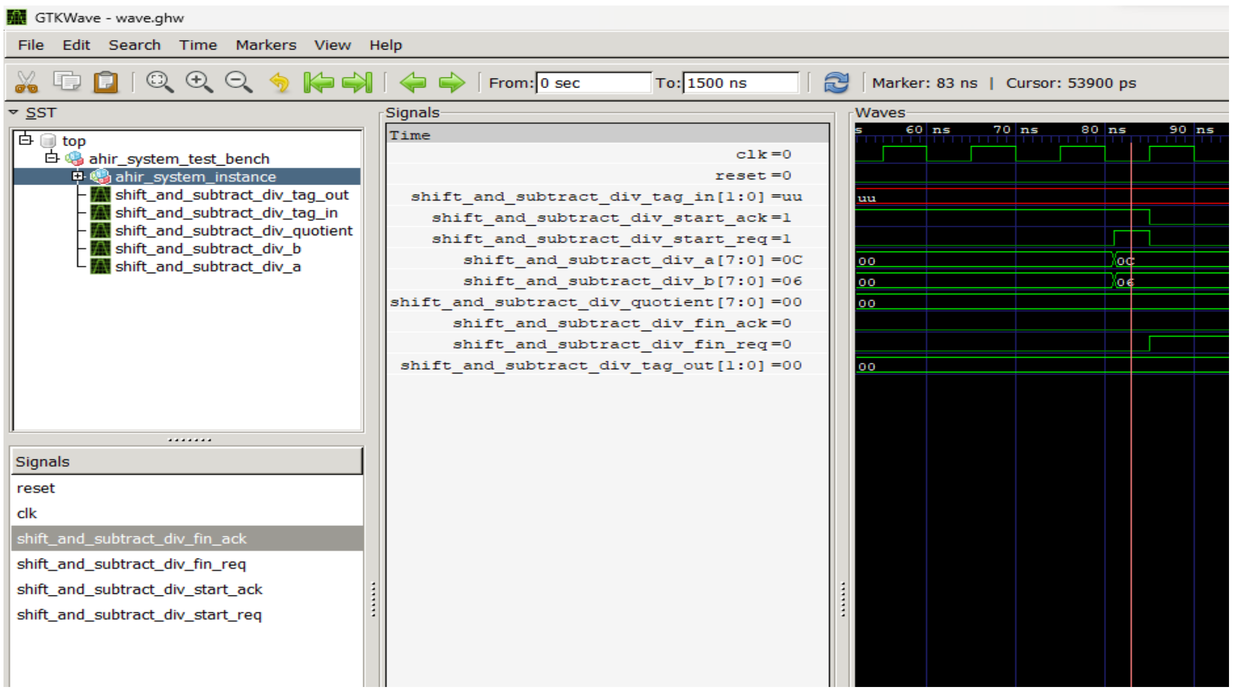
Waveform



GTKWave - wave.ghw
File Edit Search Time Markers View Help
From: 0 sec To: 1500 ns
Marker: 528 ns | Cursor: 500 ps
SST
top
ahir_system_test_bench
ahir_system_instance
shift_and_subtract_div_tag_out
shift_and_subtract_div_tag_in
shift_and_subtract_div_quotient
shift_and_subtract_div_b
shift_and_subtract_div_a
Signals
reset
clk
shift_and_subtract_div_fin_ack
shift_and_subtract_div_fin_req
shift_and_subtract_div_start_ack
shift_and_subtract_div_start_req
Waves
100 ns
200 ns
300 ns
400 ns
500 ns

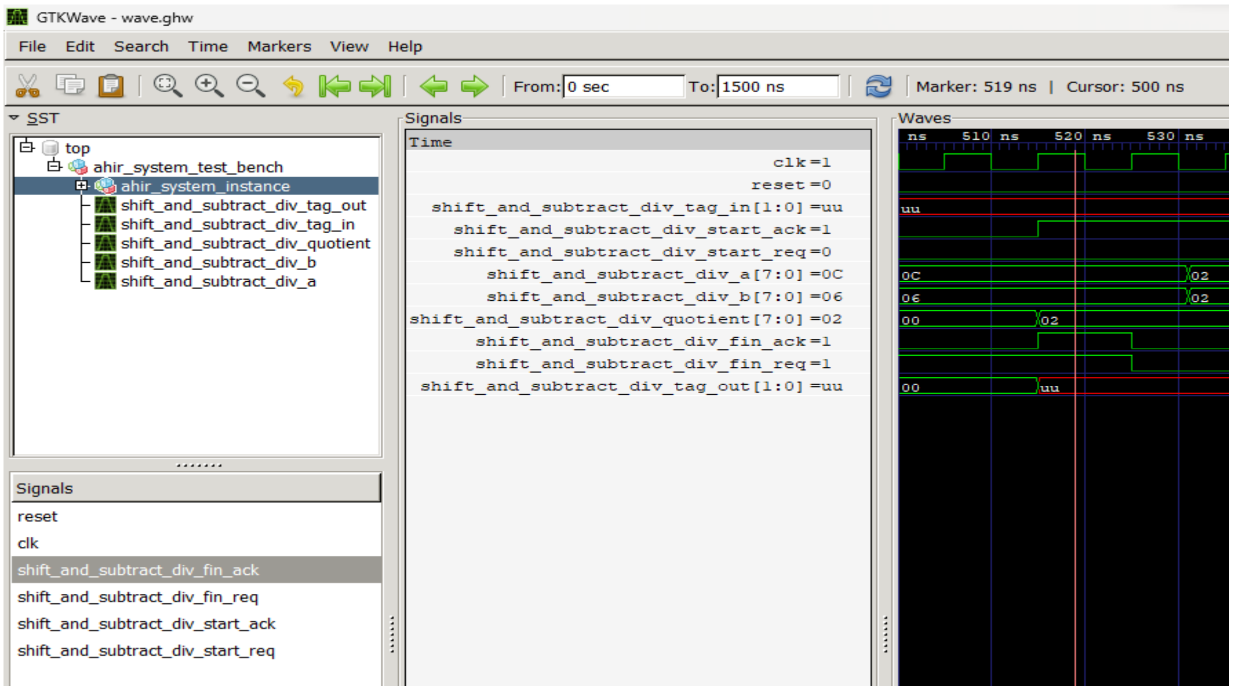
```
Time
clk=1
reset=0
shift_and_subtract_div_tag_in[1:0]=uu
shift_and_subtract_div_start_ack=1
shift_and_subtract_div_start_req=0
shift_and_subtract_div_a[7:0]=0C
shift_and_subtract_div_b[7:0]=06
shift_and_subtract_div_quotient[7:0]=02
shift_and_subtract_div_fin_ack=0
shift_and_subtract_div_fin_req=0
shift_and_subtract_div_tag_out[1:0]=uu
```

It can be seen that, when start_ack and start_req are 1, the system takes the input.


GTKWave - wave.ghw
File Edit Search Time Markers View Help
From: 0 sec
To: 1500 ns
Marker: 83 ns | Cursor: 53900 ps
SST
top
ahir_system_test_bench
ahir_system_instance
shift_and_subtract_div_tag_out
shift_and_subtract_div_tag_in
shift_and_subtract_div_quotient
shift_and_subtract_div_b
shift_and_subtract_div_a
Signals
reset
clk
shift_and_subtract_div_fin_ack
shift_and_subtract_div_fin_req
shift_and_subtract_div_start_ack
shift_and_subtract_div_start_req
Waves
60 ns
70 ns
80 ns
90 ns
uu
00
0C
00
06
00
00

```
Time
                                 clk=0
                               reset=0
     shift_and_subtract_div_tag_in[1:0]=uu
        shift_and_subtract_div_start_ack=1
        shift_and_subtract_div_start_req=1
            shift_and_subtract_div_a[7:0]=0C
            shift_and_subtract_div_b[7:0]=06
 shift_and_subtract_div_quotient[7:0]=00
          shift_and_subtract_div_fin_ack=0
          shift_and_subtract_div_fin_req=0
  shift_and_subtract_div_tag_out[1:0]=00
```

It can be seen that, When fin_ack and fin_req are 1, the system gives the output.


GTKWave - wave.ghw
File Edit Search Time Markers View Help
From: 0 sec
To: 1500 ns
Marker: 519 ns | Cursor: 500 ns
SST
top
ahir_system_test_bench
ahir_system_instance
shift_and_subtract_div_tag_out
shift_and_subtract_div_tag_in
shift_and_subtract_div_quotient
shift_and_subtract_div_b
shift_and_subtract_div_a
Signals
reset
clk
shift_and_subtract_div_fin_ack
shift_and_subtract_div_fin_req
shift_and_subtract_div_start_ack
shift_and_subtract_div_start_req
Waves
ns
510 ns
520 ns
530 ns
uu
0C
02
06
02
00
02
00
uu

```
Time
                                 clk=1
                               reset=0
     shift_and_subtract_div_tag_in[1:0]=uu
        shift_and_subtract_div_start_ack=1
        shift_and_subtract_div_start_req=0
            shift_and_subtract_div_a[7:0]=0C
            shift_and_subtract_div_b[7:0]=06
 shift_and_subtract_div_quotient[7:0]=02
          shift_and_subtract_div_fin_ack=1
          shift_and_subtract_div_fin_req=1
  shift_and_subtract_div_tag_out[1:0]=uu
```