دانشگاه صنعتی خواجه نصیرالدین طوسی

بنام خداوند جان و خرد

هم‌طراحی سخت‌افزار نرم‌افزار

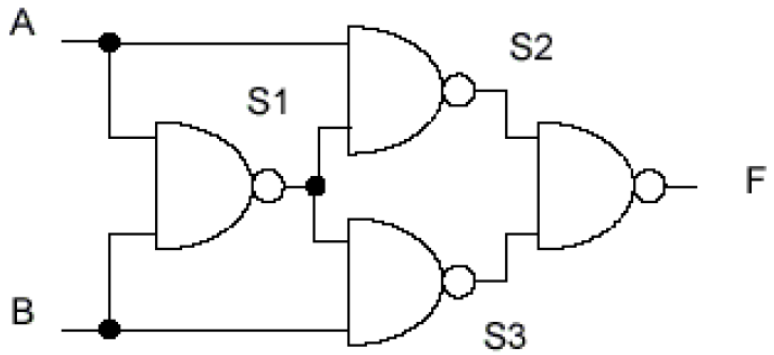
جلسه یازدهم: توصیف سیستم-زبان SystemC -۴

ارائه‌دهنده: آتنا عبدی
a_abdi@kntu.ac.ir

مباحث این بخش

- توصیف یک سیستم (System Specification)
- مدل‌های محاسباتی
- معماری‌ها
- زبان‌های توصیف
  - اشنایی با زبان توصیف سیستم SystemC

پیاده‌سازی یک سیستم ساده در SystemC



- پیاده‌سازی گیت XOR با استفاده از گیت‌های NAND
  - سیستم متشکل از زیرسیستم (گیت NAND)
    - گیت NAND ترکیبی است و خروجی کاملا تابع ورودی است
    - پیاده‌سازی پروسه با تابع SC_METHOD
    - تنظیم لیست حساسیت نسبت به ورودی‌ها

درستی‌سنجی پیاده‌سازی در SystemC

```
#include "systemc.h"
SC_MODULE(stim) {
sc_out<bool> A, B;
sc_in<bool> Clk;
void StimGen() {
A.write(false);
B.write(false);
wait();
A.write(false);
B.write(true);
wait();
A.write(true);
B.write(false);
wait();
A.write(true);
B.write(true);
wait();
sc_stop();}
SC_CTOR(stim) {
SC_THREAD(StimGen);
sensitive << Clk.pos();
}};
```

- پس از ساخت، لازم است سیستم درستی‌سنجی شود
  - با نوشتن Test bench
    - دو بخش دارد
      - Monitor
      - Stim/driver
    - ماژولی است که پروسه از نوع thread دارد
    - الگوهای مختلف مانند حساسیت به کلاک
    - در اینجا، تغییر الگوها در لبه بالارونده کلاک

درستی‌سنجی پیاده‌سازی در SystemC

```
#include "systemc.h"
#include <iostream>
#include <iomanip>
SC_MODULE(mon){
sc_in<bool> A, B, F;
sc_in<bool> Clk;
void monitor(){
cout << std::setw(10) << "Time";
cout << std::setw(2) << "A";
cout << std::setw(2) << "B";
cout << std::setw(2) << "F" << endl;
while (true){
cout << std::setw(10) <<
sc_time_stamp();
cout << std::setw(2) << A.read();
cout << std::setw(2) << B.read();
cout << std::setw(2) << F.read() <<
endl;
wait(); // wait for 1 clock cycle}}
SC_CTOR(mon){
SC_THREAD(monitor);
sensitive << Clk.pos();}};
```

- ماژول Monitor
- ساخت خروجی تست برای نمایش
- حالت متنی

تجمیع پیاده‌سازی در SystemC

```
#include "systemc.h"
#include "stim.h"
#include "exor2.h"
#include "mon.h"
int sc_main(int argc, char* argv[]) {
sc_signal<bool> ASig, BSig, FSig;
sc_clock TestClk("TestClock", 10,
SC_NS, 0.5, 1, SC_NS);
stim Stim1("Stim");
Stim1.A(ASig);
Stim1.B(BSig);
Stim1.Clk(TestClk);
exor2 DUT("exor2");
DUT.A(ASig);
DUT.B(BSig);
DUT.F(FSig);
mon Monitor1("Monitor");
Monitor1.A(ASig);
Monitor1.B(BSig);
Monitor1.F(FSig);
Monitor1.Clk(TestClk);
sc_start(); // run forever
return 0; }
```

- ماژول Main
- بالاترین سطح توصیف و تجمیع تمامی اجزا

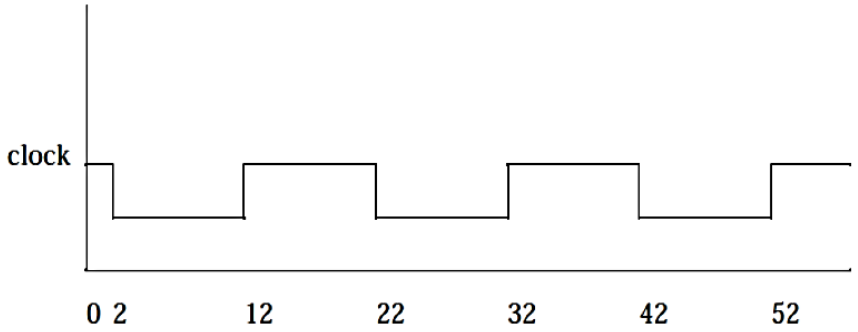

```
Microsoft Visual Studio Debug Console

      SystemC 2.3.3-Accellera --- Sep  5 2020 21:52:45
      Copyright (c) 1996-2018 by all Contributors,
      ALL RIGHTS RESERVED
    Time A B F
     0 s 0 0 0
     1 ns 0 0 0
    11 ns 0 1 1
    21 ns 1 0 1
    31 ns 1 1 0

Info: /OSCI/SystemC: Simulation stopped by user.

C:\Users\Athena\source\repos\Project1\Debug\Project1.exe (process 9488) exited with code 0.
Press any key to close this window . . .
```

نکته تکمیلی

- تعریف کلاک

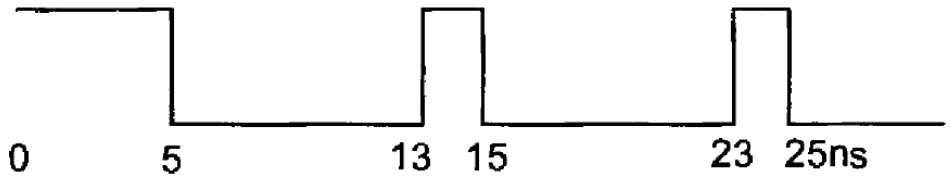
sc_clock TestClk(“Name", period (@ time unit), Duty Cycle, First edge occurrence (@ time unit), First value);

```
Example: sc_clock TestClk("TestClock", 20, 0.5, 2, true);
```



- بهتر است در ابتدای شبیه‌سازی لبه ایجاد نکنیم
  - منجر به عدم قطعیت و جواب نادرست می‌شود

```
Example: sc_clock TestClk("TestClock", 10, 0.2, 5, true);
```



شبیه‌سازی خروجی سیستم توسط شکل موج

- کتابخانه systemC قابلیت نمایش شکل موج را دارد
  - فعال‌سازی این قابلیت از طریق افزدون دستوراتی در تابع main
    - ایجاد یک فایل Trace و افزودن سیگنال‌ها و پورت‌ها به آن
  - تولید شکل‌موج خروجی در یکی از فرمت‌های
    - VCD، WIF ASCII یا ISDB
  - نمایش شکل‌موج در ابزارهای
    - GTKWave، SystemC_Win

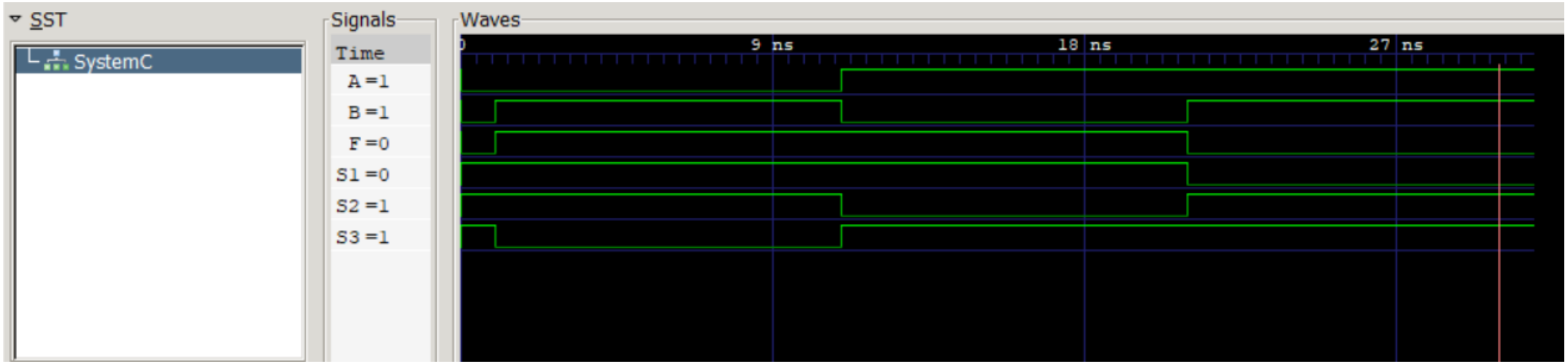
شبیه‌سازی خروجی سیستم توسط شکل موج

```
int sc_main(int argc, char* argv[]){
sc_signal<bool> ASig, BSig, FSig;
sc_clock TestClk …
// here add Instance of stim
exor2 DUT("exor2");
DUT.A(ASig);
DUT.B(BSig);
DUT.F(FSig);
// here add Instance of mon
sc_trace_file* Tf =
sc_create_vcd_trace_file("trace_xor");
Tf->set_time_unit(1, SC_NS);
sc_trace(Tf, ASig, "A");
sc_trace(Tf, BSig, "B");
sc_trace(Tf, FSig, "F");
sc_trace(Tf, DUT.S1, "S1");
sc_trace(Tf, DUT.S2, "S2");
sc_trace(Tf, DUT.S3, "S3");
sc_start(); // run forever
sc_close_vcd_trace_file(Tf);
return 0;}
```

- مثال گیت XOR
- تغییر تابع main
- افزودن دستورات ایجاد فایل trace

شبیه‌سازی خروجی سیستم توسط شکل موج

- مثال گیت XOR
- نمایش فایل vcd خروجی در GTKWave



پیاده‌سازی یک سیستم واقعی در SystemC

- مدلسازی کنترلر چراغ راهنمایی
  - چراغ مسیر اصلی (شمالی-جنوبی) به‌صورت پیش‌فرض سبز است
  - با ورود خودرو به مسیر فرعی (شرقی-غربی) حسگرهایی فعال می‌شوند
    - چراغ مسیر اصلی: سبز-> زرد -> قرمز
      - با تاخیر مشخص شده
    - چراغ مسیر فرعی: قرمز -> سبز
      - بازگشت به قرمز پس از زمان مشخص

مدلسازی کنترلر چراغ راهنمایی در SystemC

```
// traff.h
#include "systemc.h"
SC_MODULE(traff) {
// input ports
sc_in<bool> roadsensor;
sc_in<bool> clock;
// output ports
sc_out<bool> NSred;
sc_out<bool> NSyellow;
sc_out<bool> NSgreen;
sc_out<bool> EWred;
sc_out<bool> EWyellow;
sc_out<bool> EWgreen;
void control_lights();
// Constructor
SC_CTOR(traff) {
SC_THREAD(control_lights);
// Thread
sensitive << roadsensor;
sensitive << clock.pos();}
};
```

- ورودی‌های سیستم:
  - حسگرهای مسیر شرقی-غربی و سیگنال کلاک
- خروجی‌های سیستم:
  - چراغ‌های سبز، زرد و قرمز مسیر اصلی و مسیر فرعی

مدلسازی کنترلر چراغ راهنمایی در SystemC

```
// traff.cpp
#include "traff.h"
void traff::control_lights() {
// init
NSred = false;
NSyellow = false;
NSgreen = true;
EWred = true;
EWyellow = false;
EWgreen = false;
while (true) {
while (roadsensor == false)
wait();
NSgreen = false;// road sensor triggered
NSyellow = true; // set NS to yellow
NSred = false;
for (i = 0; i < time_a; i++)
wait();
NSgreen = false; // yellow interval over
NSyellow = false; // set NS to red
NSred = true; // set EW to green
EWgreen = true;
EWyellow = false;
EWred = false;
```

- طراحی زمان تغییر چراغ‌ها و روال کنترل

```
// Determined Time for EW path
for (i = 0; i < time_b; i++)
wait();
NSgreen = false; // times up for EW green
NSyellow = false; // set EW to yellow
NSred = true;
EWgreen = false;
EWyellow = true;
EWred = false;
for (i = 0; i < time_a; i++)
// times up for EW yellow
wait();
NSgreen = true; // set EW to red
NSyellow = false; // set NS to green
NSred = false;
EWgreen = false;
EWyellow = false;
EWred = true;
for (i = 0; i < time_b; i++) // wait one more long
wait(); }} // interval before allowing a sensor again
```

مدلسازی کنترلر چراغ راهنمایی در SystemC

- کد تست (Test bench)

```
// test.h
#include "systemc.h"
SC_MODULE(test_bench) {
sc_out<bool> roadsensor;
sc_in<bool> Clk;
void process_1(){
roadsensor = false;
wait();
roadsensor = true;
wait(2);
roadsensor = false;
wait();
sc_stop();}
SC_CTOR(test_bench) {
SC_THREAD(process_1);
sensitive << clock.pos();}
};
```

به اندازه دوبار فعال شدن شرط حساسیت (لبه بالارونده کلاک)

مدلسازی کنترلر چراغ راهنمایی در SystemC

- کد main

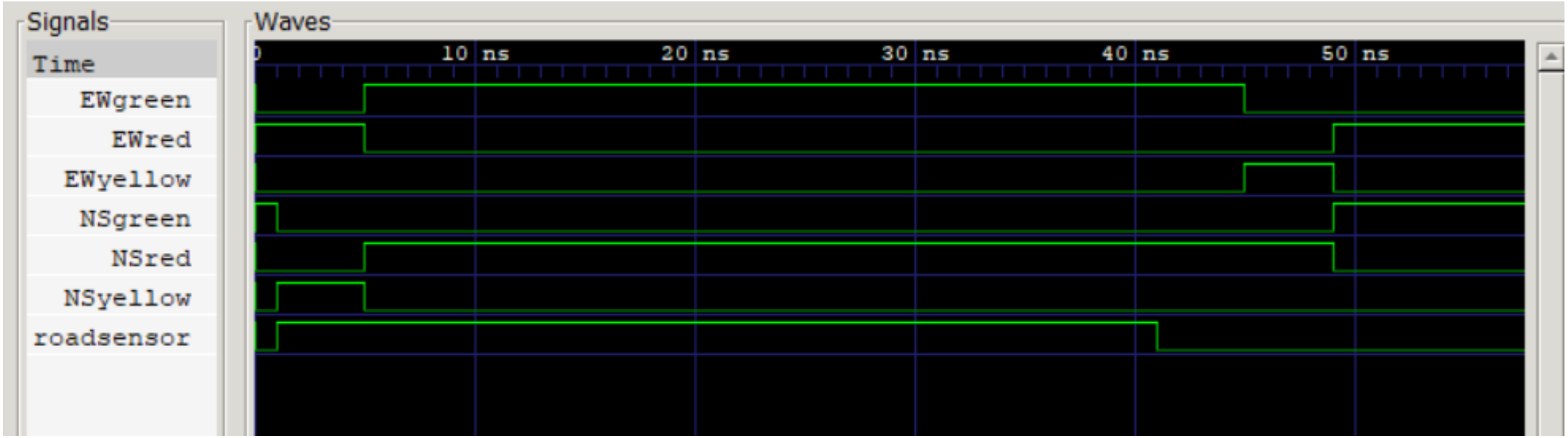
```
#include "systemc.h"
#include "test.h"
#include "traff.h"
int sc_main(int, char* [])
{sc_signal< bool > rSig;
sc_signal< bool > NSred_sig, NSyellow_sig, NSgreen_sig,
EWred_sig, EWyellow_sig, EWgreen_sig;
sc_clock TestClk("TestClock", 20, SC_NS, 0.5, 1, SC_NS);
-- instance of trff
-- instance of test_bench
sc_trace_file* tf;
sc_trace_file* Tf = sc_create_vcd_trace_file("trace_traff3");
Tf->set_time_unit(1, SC_NS);
sc_trace(Tf, rSig, "roadsensor");
sc_trace(Tf, NSred_sig, "NSred");
sc_trace(Tf, NSyellow_sig, "NSyellow");
sc_trace(Tf, NSgreen_sig, "NSgreen");
sc_trace(Tf, EWred_sig, "EWred");
sc_trace(Tf, EWyellow_sig, "EWyellow");
sc_trace(Tf, EWgreen_sig, "EWgreen");
sc_start(); // run forever
sc_close_vcd_trace_file(Tf);
return 0;}
```

مشابه مثال قبل

مواردی که در شکل موج خروجی نمایش داده می‌شوند

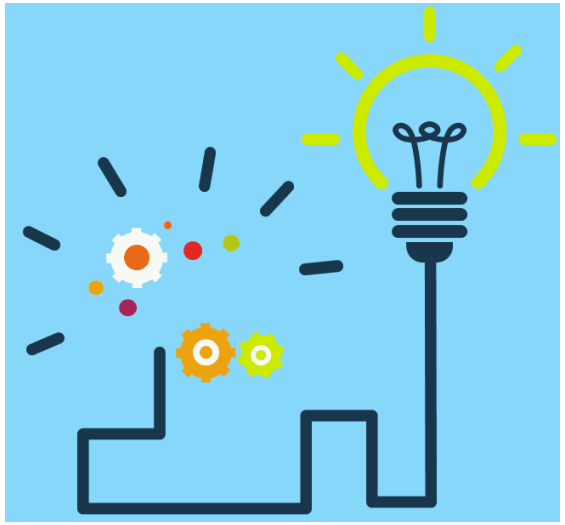
مدلسازی کنترلر چراغ راهنمایی در SystemC

- نتیجه نهایی (فایل trace)



مباحثی که این جلسه آموختیم

- توصیف سیستم و زبان
- آشنایی با زبان SystemC
- اعتبارسنجی متنی و نمایشی
- مثال مدلسازی سیستم



مباحث جلسه آینده

- شروع فرایند سنتز توأم
- آشنایی با روال سنتز توأم و اجزا و پروسه‌های آن