# Digital IC Design

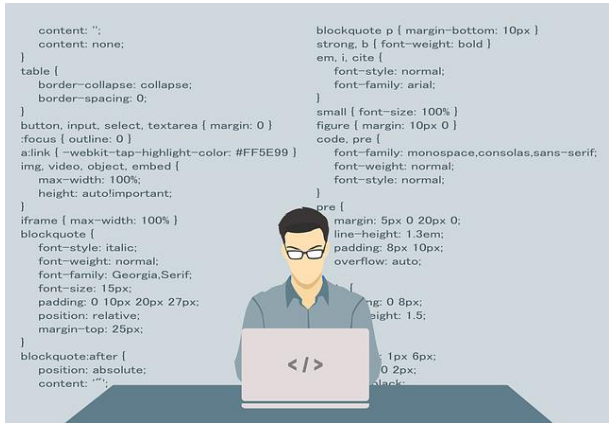
## Week4: Simulation using Cadence Xcelium™

2024 | Nineplus Infotech X Anseong Polytechnic University
*Sr Research Engineer, William Woo*

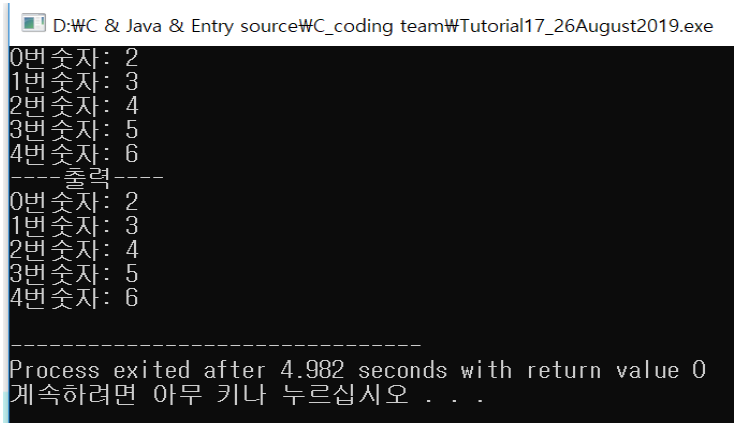
cadence

# 1. Simulation Overview

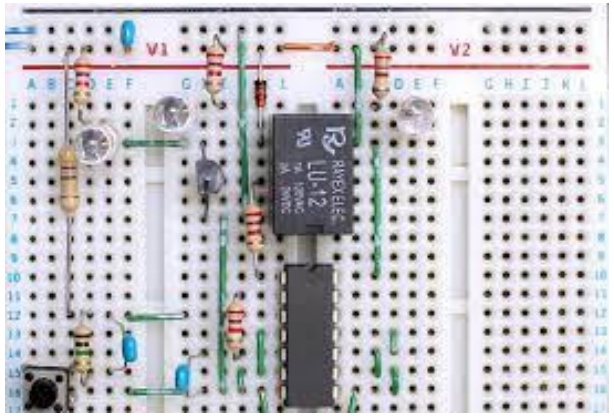
# Imagine your conventional debugging

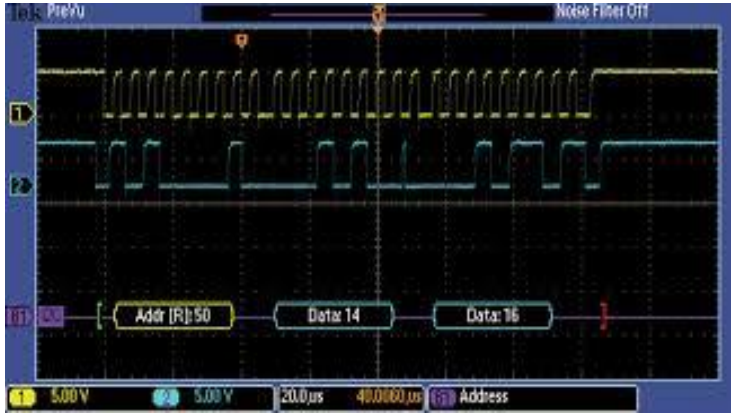


***Design***



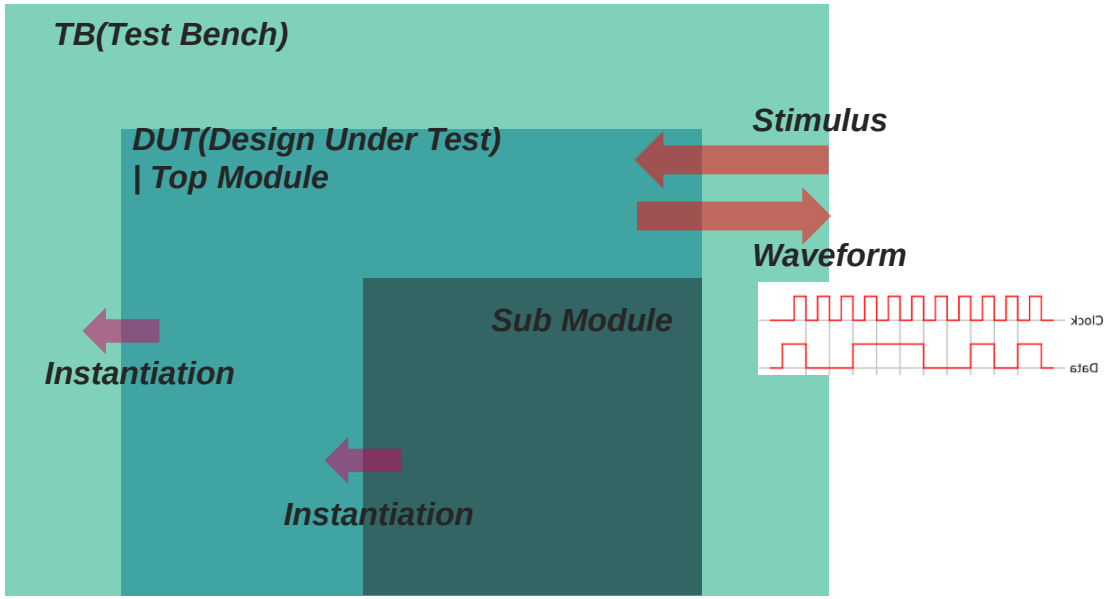

***Verifying & Debugging***

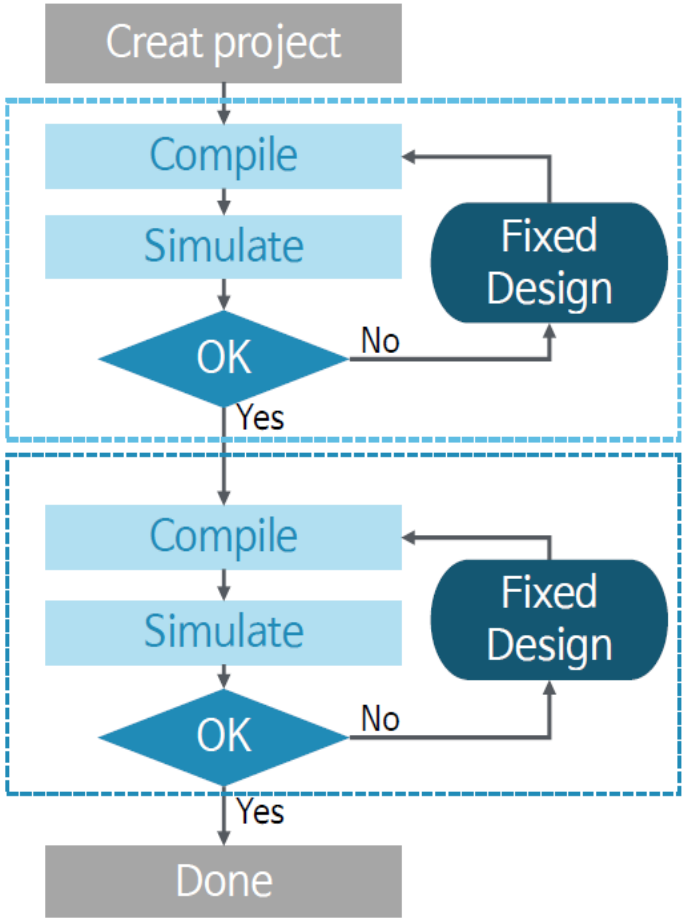




# Simulation Flow

- Simulation
  - 설계 블록(DUT)에 Stimulus를 적용
  - Waveform을 확인

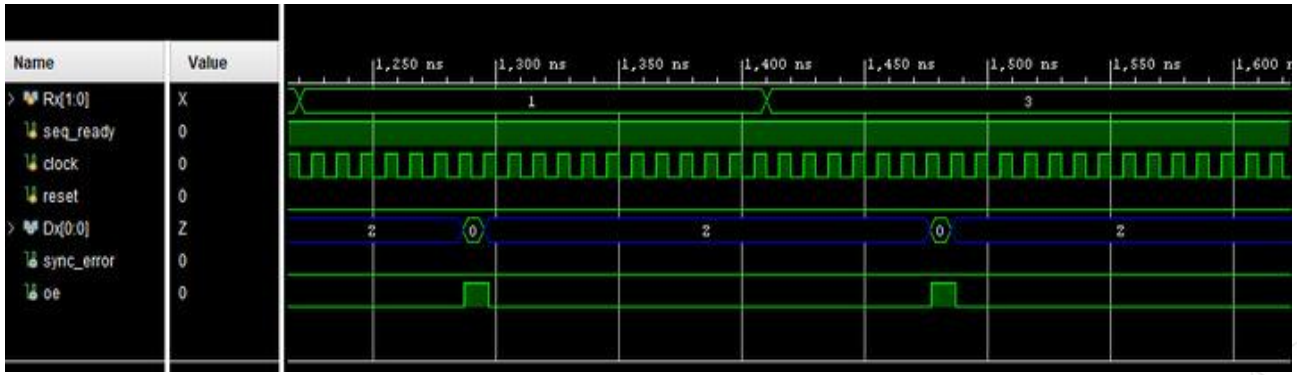


# Types of Simulation

- Functional Simulation
  - 로직의 동작을 확인
  - Gate/propagation delay는 고려하지 않음
  - 검증이 간단하고 빠름


- Timing Simulation
  - 로직과 wire의 delay를 설명
  - Function Simulation보다 현실적임
  - 검증이 복잡하고 느림



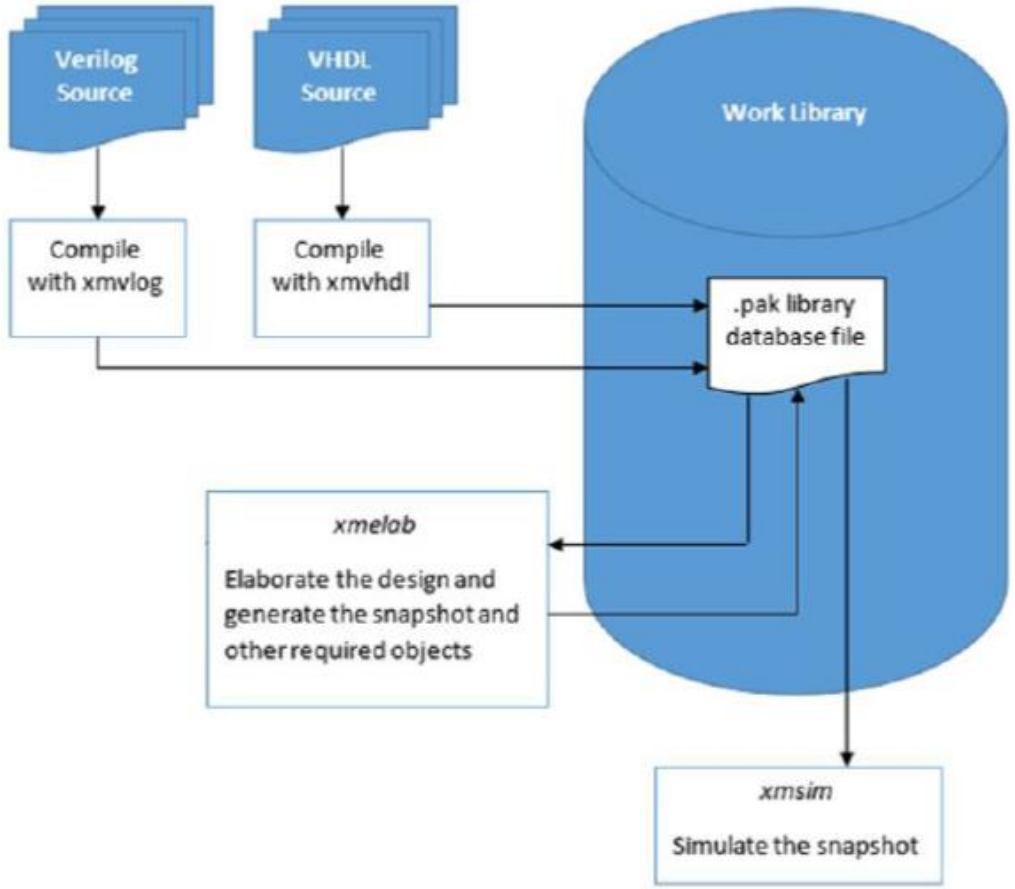
# Simulation Process

- Simulation을 준비하는 단계
- 모듈과 인스턴스 결합
- 신호 연결

***Compile*** → ***Elaboration*** → ***Simulation***

- 문법 체크
- Simulation이나 Synthesis에 필요한 파일 생성

- Design module의 실행
- Signal/Net에서의 변화 값을 계산하여 출력



# 2. Cadence Xcelium™ Simulator

# Simulation Step[s] in Xcelium® Simulator

***Multi Step Simulation***

**Design Setup**: library & setting file

**xmvlog** : compile source file

**xmelab**: elaborate the design

**xmsim**: start simulation

***Single Step Simulation***

**xrun**: compile + elaborate+ simulation

# Multi-step Simulation(1): Design Setup

**Design Setup**: library & setting file

```
// cds.lib

define {alias} {LIBRARY_NAME}
include {OTHER_REFERENCED_CDS.LIB}
…
→ 설계 및 simulation에서 사용/생성되는 library관리 파일
```

```
// hdl.var
define WORK {alias}
define {SIM_OPTION_VAR} -option
…

→ compile, elaboration, simulation에서 사용되는 옵션 및 설정 관리 파일
```

***Referenced in each step as environmental files***

**xmvlog** : compile source file

**xmelab**: elaborate the design

**xmsim**: start simulation

# Multi-step Simulation(Compile)

**xmvlog** : compile source file

$ xmlvog –message –linedebug mux.v

- -message OR –MESS: compile 진행과정/결과를 모듈별로 출력
- -linedebug: verilog 코드에 breakpoint를 적용하여 code line별로 debugging을 수행할 수 있도록 함

- L.C:V

Library.Cell:View

```
[toolsetup@nprnd lab1_mux]$ xmvlog -MESS -linedebug mux.v
xmvlog: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
file: mux.v
        module mux4_to_1.mux:vlog
                errors: 0, warnings: 0
```

```
[toolsetup@nprnd lab1_mux]$ xmvlog -MESS -linedebug mux_test.v
xmvlog: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
file: mux_test.v
        module mux4_to_1.mux_test:vlog
                errors: 0, warnings: 0
```

***$ xmls –all: Compile확인***

```
[toolsetup@nprnd lab1_mux]$ xmls -all
xmls: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
        module mux4_to_1.mux:vlog (VST)
        module mux4_to_1.mux_test:vlog (VST)
```

# Multi-step Simulation(Elaboration)

**xmelab**: elaborate the design

$ xmelab –message –access RWC mux_test

- -message OR –MESS: compile 진행과정/결과를 모듈별로 출력
- -access RWC: compile된 verilog파일에 RWC권한을 부여하여 파형을 확인할 수 있도록 함

```
[toolsetup@nprnd lab1_mux]$ xmelab -MESS -access RWC mux_test
xmelab: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
        Elaborating the design hierarchy:
                Caching library 'mux4_to_1' ....... Done
        Building instance overlay tables: ..................... Done
        Generating native compiled code:
                mux4_to_1.mux:vlog <0x026181f9>
                        streams:   1, words:   497
                mux4_to_1.mux_test:vlog <0x5f71d92a>
                        streams:   5, words:  5073
        Building instance specific data structures.
        Loading native compiled code:     ..................... Done
        Design hierarchy summary:
                                Instances  Unique
                Modules:                2       2
                Registers:              4       4
                Scalar wires:           4       -
                Always blocks:          1       1
                Initial blocks:         1       1
                Pseudo assignments:     3       -
                Simulation timescale:  1ns
        Writing initial simulation snapshot: mux4_to_1.mux_test:vlog
```

```
[toolsetup@nprnd lab1_mux]$ xmls -all
xmls: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
        module mux4_to_1.mux:vlog (VST)
        module mux4_to_1.mux:vlog (SIG) <0x026181f9>
        module mux4_to_1.mux:vlog (COD) <0x026181f9>
        module mux4_to_1.mux_test:vlog (VST)
        module mux4_to_1.mux_test:vlog (SIG) <0x5f71d92a>
        module mux4_to_1.mux_test:vlog (COD) <0x5f71d92a>
        snapshot mux4_to_1.mux_test:vlog (SSS)
```

# Multi-step Simulation(Simulation: no-gui)

**xmsim**: start simulation

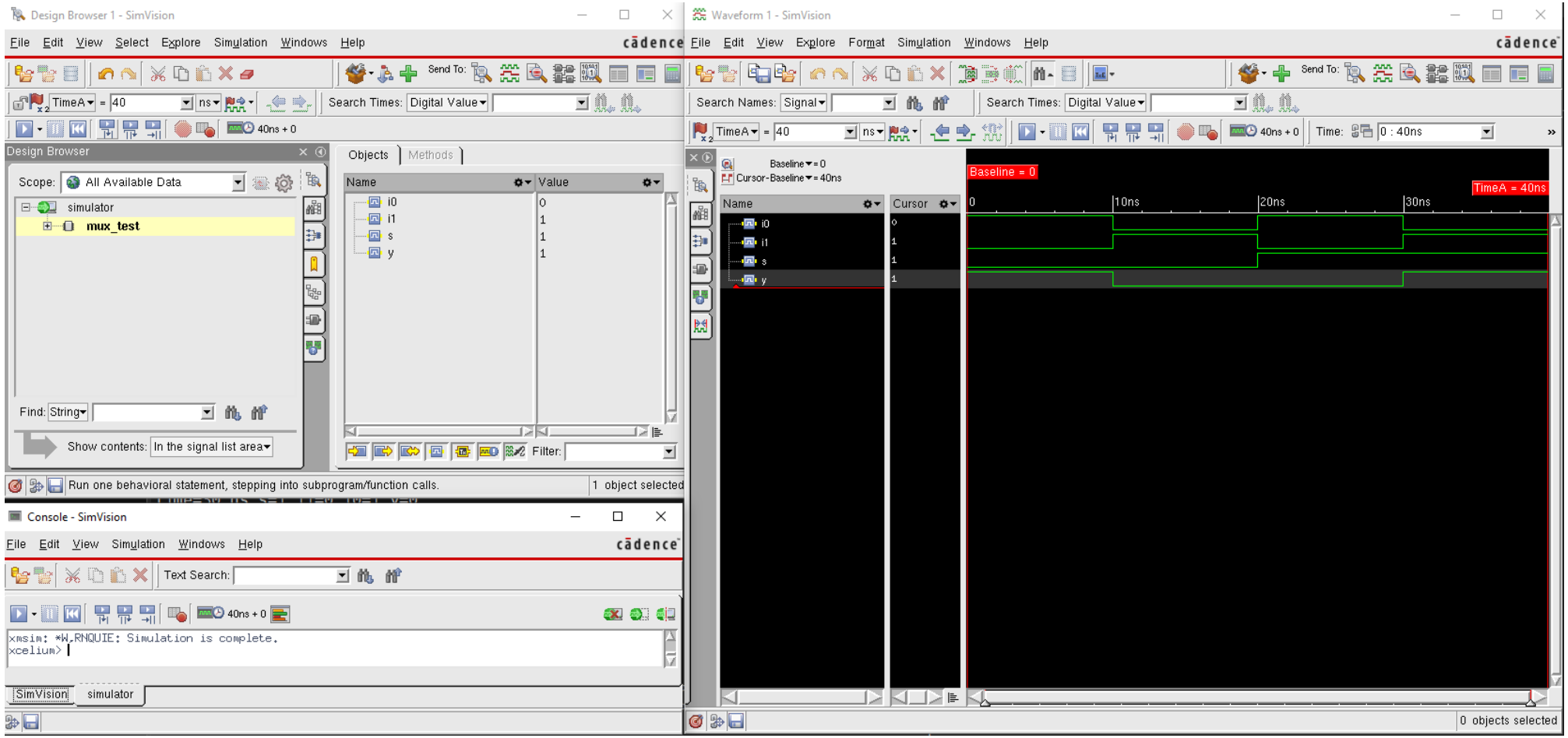
$ xmsim –message mux_test -gui

- -message OR –MESS: compile 진행과정/결과를 모듈별로 출력
- -gui: gui tool(simvision)을 구동함

```
[toolsetup@nprnd lab1_mux]$ xmsim mux_test
xmsim: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
xcelium> run
time=10 ns s=0 i1=0 i0=1 y=1
time=20 ns s=0 i1=1 i0=0 y=0
time=30 ns s=1 i1=0 i0=1 y=0
time=40 ns s=1 i1=1 i0=0 y=1
xmsim: *W,RNQUIE: Simulation is complete.
xcelium> exit
```

***xmsim: no-gui mode***

# Multi-step Simulation(Simulation: gui)



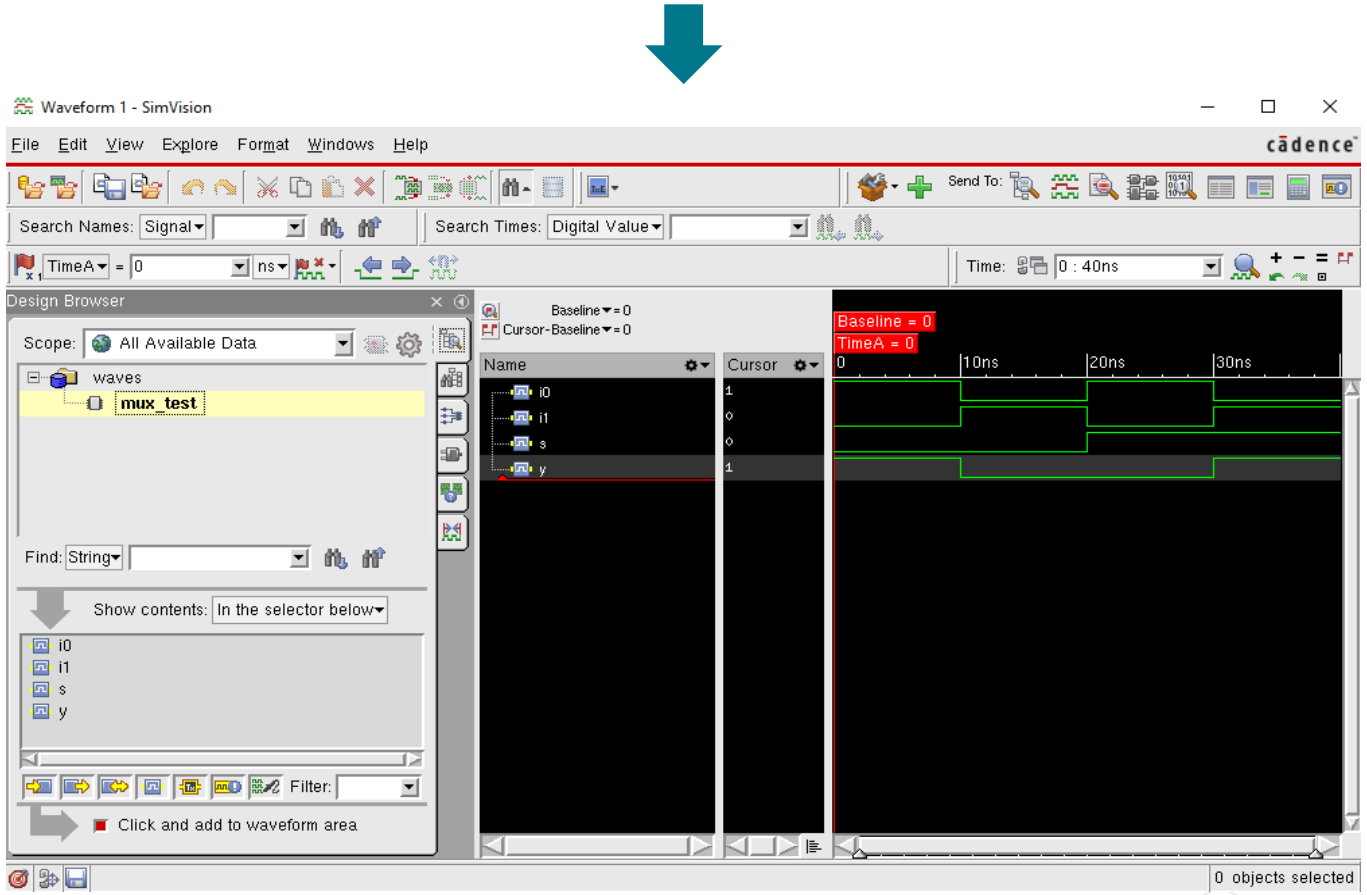
# Multi-step Simulation(Simulation:PPE)

**PPE**: Post Processing Environment

$ simvison waves.shm

- PPE모드에서는 이미 완료된 simulatio결과에 대해 파형을 관찰하는 경우에 사용
- 이미 생성된 waves.shm database에 access

```
[toolsetup@nprnd lab1_mux]$ simvision waves.shm
simvision: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
txe: 23.03-s003: (c) Copyright 1995-2023 Cadence Design Systems, Inc.
```



# LAB

1. DUT design applying module instantiation
2. Write test bench according to truth table on the data sheet
3. 3-step simulation
4. Make file automation