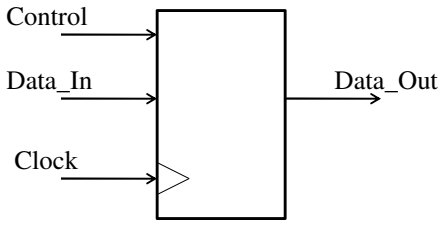
Pollard's Introduction to VHDL (Session 2)

Register Behavior



Entity Statement

- Identifies interface mechanism
- Can also identify constant values
- Interface elements - signals
- No work can occur in entity
- Checking and passive routines allowed

Architecture Body

- Holds description of the work
- Has access to signals in port, information in generic of entity
- Local elements identified in declaration area
- Description can be behavioral or structural
- Communication only through port

Architecture Syntax

```
architecture ARCH_NAME of ENT_NAME is
  { declaration area }
begin
  {specification of parallel activities }
end {architecture}{ARCH_NAME};
```

Concurrent Statement

- Block statement
- Process statement
- Component instantiation
- Generate statement
- Concurrent signal assignment statement
- Concurrent assertion statement
- Concurrent procedure call statement

Process Statement

- Concurrent statement
- Activity in process - sequential
- Delay mechanism user selectable
  - sensitivity list - comma separated list of signals
  - event on any signal in list causes process activity
  - wait statements in process body
- Activity visited once on startup

Process Statement Syntax

```
PROC_NAME: process ( SEN_LIST ) is

  { declaration area }
begin
  {sequential statement}*
end process {PROC_NAME};
```

Sequential Statements

- Signal Assignment Statement
- If Statement
- Case Statement
- Loop Statement
- Wait statement
- Assertion Statement
- Report Statement

Sequential Statements (Cont.)

- Variable assignment statement
- Procedure Call Statement
- Next Statement
- Exit Statement
- Null Statement

If Statement Syntax

```
If  BOOL_EXP then
    {sequential statement}
elsif BOOL_EXP then
    {sequential statement}
else
    {sequential statement}
end if;
```

Case Statement Syntax

```
case   SEL_EXP  is
    when  CHOICE  =>  {sequential stmt}
    when  CHOICE | CHOICE =>  {seq stmt}
    when  others  => {sequential stmt}
end case ;
```

Loop Statement Syntax

for *ID* in *RANGE* loop
    {sequential statement}
end loop;

while *BOOL_EXP* loop
    {sequential statement}
end loop;

Loop Statement Syntax (Cont.)

go to next iteration:
    next { when *BOOL_EXP* } ;

get out of loop:
    exit { when *BOOL_EXP* } ;

Wait Statement Syntax

wait on *SENSITIVITY_LIST*
until *BOOLEAN_EXP*
for *TIME_EXP* ;

wait on A, B, C for 60 ns;
wait until DATA = FE for 10 ns;