Teaching Digital Design with HDL

M. E. Parten
Department of Electrical Engineering
Texas Tech University
Lubbock, Texas 79409-3102

Abstract

This paper describes the use of hardware descriptive languages (HDL) in an introductory, sophomore level digital design course in electrical engineering. HDL is integrated with the other basic tools in design and simulation of combinational and sequential systems. A number of examples are given.

Introduction

The use of hardware descriptive languages (HDL) to design digital systems is becoming increasingly common in industry. However, most introductory digital courses in universities do not use hardware descriptive language to any great degree. In the past, most hardware descriptive language software packages were very expensive and frequently only ran on workstations or larger computers. However, recently, there has been an increase in the availability of inexpensive and even free HDL software. This development makes the use of HDL in introductory courses possible and even advantageous. Since most students, these days, already have experience with computers and some programming, HDL is more natural for them than classical digital design techniques.

Frequently, when HDL is used in a digital design course, it is taught as a separate topic instead of integrating it with other basic concepts. In addition, it usually doesn't occur until after combinational and sequential circuits have been covered. However, the newer software packages available today allow HDL to be used as a standard tool in the design and simulation of digital systems. This applies not only to sequential systems but also simple combinational circuits.

Most introductory digital design courses teach basic digital logic simplification using basic Boolean algebra and Karnaugh maps, as evidenced by many popular textbooks available today[1,2] . However, Karnaugh maps become more difficult to use with more than four variables. Variable entry mapping is frequently used[3] for higher numbers of variables, but it is difficult to know if it is a true minimum solution. Numerous computer programs have been written that enable simplification of larger systems using variations of the Quine-McCluskey method and other techniques. However, these are frequently past over in many digital courses.

Sequential systems lend themselves to algorithmic type analysis also. And over the past several years, a number of hardware descriptive languages (HDL) have been developed that allow algorithmic design of digital systems with the software program performing the reduction and fitting of the problem to a specific set of hardware. This is done for programmable logic arrays and for integrated circuit design. Although there are many such hardware descriptive languages, the standard for IC design is called VHDL. Other HDLs are usually specific to the vendor of specific chips.

Although digital design using HDL has been around for some time and is well entrenched in industry, HDL is usually only taught at universities in more advanced or elective courses. One of the problems with teaching HDL has been the software has relatively expensive and not available to students directly. However, there are now software packages available to students to run on standard PCs that include powerful hardware descriptive languages[4]. Although these packages may not posses all of the tools available for VHDL, they do allow students to learn the basic constructs of hardware descriptive languages and use the tools to design complex systems.

Another problem is the lack of textbooks. There are a number of textbooks on VHDL[5,6], and some that include VHDL[7] after completing standard design.

An Introductory Course

In the Department of Electrical Engineering at Texas Tech University, students are introduced to HDL in their first course in digital systems, EE 2372. There are no prerequisites for the course and it is usually taken the second semester of the freshman year. The outline for the course are given below.

EE 2372

Textbook:	*Modern Digital System Design by Cheung and Bredeson (C)* *Pspice for Windows Vol. II by Goody* *Notes*
References:	*A number of digital books are also available on reserve in the library listed under the EE labs.*
Programs:	*Pspice, PLsyn*

Week	***Topic***	***Book***
1	*Number Systems*	*(C) Ch 1*
2	*Boolean Algebra*	*(C) Ch 2*
3-5	*Combinational Logic with Intro. To HDL*	*(C) Ch 3* *Plsyn & Pspice*
6,7	*Sequential Machines*	*(C) Ch. 5* *Plsyn*
8-10	*Programmable logic devices*	*(C) Ch 6* *Plsyn*
11	*MSI Devices*	*(C) Ch 4* *Pspice, Plsyn*
12,13	*Digital system examples*	*Notes* *Pspice & Plsyn*
14	*Ladder Logic*	*Notes*

Plsyn is MicroSim's version of a hardware descriptive language. Wherever Plsyn appears in the outline is where HDL is utilized.

Most hardware descriptive languages have a number of ways to present logical statements. As an example a standard seven segment decoder can be represented in a number of different truth table forms, as indicated below.

```
PROCEDURE Sev_Seg_c( INPUT D[3..0];
    OUTPUT Oa, Ob, Oc, Od, Oe, Of, Og);

TRUTH_TABLE
D[3..0] :: Oa, Ob, Oc, Od, Oe, Of, Og;
"-------------------------------------
  0   :: 1,  1,  1,  1,  1,  1,  0;
  1   :: 0,  1,  1,  0,  0,  0,  0;
  2   :: 1,  1,  0,  1,  1,  0,  1;
  3   :: 1,  1,  1,  1,  0,  0,  1;
  4   :: 0,  1,  1,  0,  0,  1,  1;
  5   :: 1,  0,  1,  1,  0,  1,  1;
  6   :: 1,  0,  1,  1,  1,  1,  1;
  7   :: 1,  1,  1,  0,  0,  0,  0;
  8   :: 1,  1,  1,  1,  1,  1,  1;
  9   :: 1,  1,  1,  1,  0,  1,  1;
 ELSE   ::.X.,.X.,.X.,.X.,.X.,.X.; "Don't Cares
END TRUTH_TABLE;

END Sev_Seg_c;
```

Figure 1. Seven Segment Display Logic

The logic variables can be set as individual variables or dimensioned variables expressed in binary or hexadecimal. The truth table form is very easy for students to follow. Logic statements can also be entered directly as can "if -- then" type algorithms as shown in Figure 2.

```
PROCEDURE Comp_4b( INPUT A[3..0],
 B[3..0]; OUTPUT AgB, AeqB, BgA );


" Declaration of Procedure for a 4-Bit Comparator

IF    A > B THEN [AgB, AeqB, BgA]=100b;
ELSIF A = B THEN [AgB, AeqB, BgA]=010b;
ELSIF A < B THEN [AgB, AeqB, BgA]=001b;
END IF;

END Comp_4b;
```

Figure 2. 4-Bit Comparator Logic

Again, this is a natural way for students to enter this type of operation. Part of the output from the program is shown below in Figure 3. As can be seen, the program generates some internal variables to simplify the expressions. This provides a great opportunity to discuss alternative implementations.

```
LOGIC:
 AgtB = { A3$$ &  ~B3$$ |  A3$$ &  Comp_4b$_4BIT_Comp$less_than$_$4 |  ~B3$$ &
Comp_4b$_4BIT_Comp$less_than$_$4 }

AeqB = { ~(  A3$$ &  ~B3$$ |  A3$$ &  Comp_4b$_4BIT_Comp$less_than$_$4 | ~B3$$ &
Comp_4b$_4BIT_Comp$less_than$_$4 |   Comp_4b$_4BIT_Comp$equal_to$_$3 ) }

 AltB = { ~( A3$$ &  ~B3$$ |   A3$$ &  Comp_4b$_4BIT_Comp$less_than$_$4 |  ~B3$$ &
Comp_4b$_4BIT_Comp$less_than$_$4 |  ~Comp_4b$_4BIT_Comp$equal_to$_$3 ) }

 Comp_4b$_4BIT_Comp$less_than$_$2 = {A0$$ & ~B0$$ | A0$$ &  $D_LO |  ~B0$$ &
$D_LO }

Comp_4b$_4BIT_Comp$less_than$_$3 = {A1$$ &  ~B1$$ |  A1$$ &
Comp_4b$_4BIT_Comp$less_than$_$2 |  ~B1$$ &   Comp_4b$_4BIT_Comp$less_than$_$2 }

Comp_4b$_4BIT_Comp$less_than$_$4 = {A2$$ &  ~B2$$ | A2$$ &
Comp_4b$_4BIT_Comp$less_than$_$3 | ~B2$$ &   Comp_4b$_4BIT_Comp$less_than$_$3 }

Comp_4b$_4BIT_Comp$equal_to$_$1 = {A0$$ & ~B0$$ | ~A0$$ & B0$$ |  A1$$ &  ~B1$$ |
~A1$$ & B1$$ }

Comp_4b$_4BIT_Comp$equal_to$_$2 = {A2$$ &  ~B2$$ |  ~A2$$ &  B2$$ | A3$$ & ~B3$$ |
~A3$$ & B3$$ }

Comp_4b$_4BIT_Comp$equal_to$_$3 = {Comp_4b$_4BIT_Comp$equal_to$_$1 |
Comp_4b$_4BIT_Comp$equal_to$_$2 }

Comp_4b$_4BIT_Comp$less_than$_$6 = {~A0$$ & B0$$ | ~A0$$ &  $D_LO |  B0$$ & $D_LO
}

Comp_4b$_4BIT_Comp$less_than$_$7 = {~A1$$ & B1$$ | ~A1$$ &
Comp_4b$_4BIT_Comp$less_than$_$6 |  B1$$ &   Comp_4b$_4BIT_Comp$less_than$_$6 }


Comp_4b$_4BIT_Comp$less_than$_$8 = {~A2$$ & B2$$ |  ~A2$$ &
Comp_4b$_4BIT_Comp$less_than$_$7 |  B2$$ &   Comp_4b$_4BIT_Comp$less_than$_$7 }
```

Figure3. 4-Bit Comparator Logic Equations

Sequential Circuits

Although it is handy to use HDL on combinational circuits, the real power of HDL is shown on sequential circuits. The algorithmic design of sequential circuits is frequently more natural to students that are familiar with programming. This allows the students to

pick up the concepts faster and gives them the capability to approach much more complex problems. As in the case of combinational circuits, a number of different input combinations are possible.

```
" GRAY1

PROCEDURE GRAY1 (INPUT clock, reset;
        OUTPUT q3, q2, q1, q0 CLOCKED_BY clock RESET_BY reset);

q3 = Q3*/Q2*/Q1*Q0 + Q3*/Q2*Q1*Q0 + Q3*/Q2*Q1*/Q0 + Q3*Q2*Q1*/Q0
        + Q3*Q2*Q1*Q0 + Q3*Q2*/Q1*Q0 + Q3*Q2*/Q1*/Q0 + /Q3*Q2*/Q1*/Q0;

q2 = Q3*Q2*Q1*Q0 + Q3*Q2*/Q1*Q0 + Q3*Q2*/Q1*/Q0 + /Q3*Q2*/Q1*/Q0
        + /Q3*Q2*/Q1*Q0 + /Q3*Q2*Q1*Q0 + /Q3*Q2*Q1*/Q0 + /Q3*/Q2*Q1*/Q0;

q1 = Q3*/Q2*Q1*/Q0 + Q3*Q2*Q1*/Q0 + Q3*Q2*Q1*Q0 + Q3*Q2*/Q1*Q0
        + /Q3*Q2*Q1*/Q0 + /Q3*/Q2*Q1*/Q0 + /Q3*/Q2*Q1*Q0 + /Q3*/Q2*/Q1*Q0;

q0 = Q3*/Q2*Q1*Q0 + Q3*/Q2*Q1*/Q0 + Q3*Q2*/Q1*Q0 + Q3*Q2*/Q1*/Q0
        + /Q3*Q2*Q1*Q0 + /Q3*Q2*Q1*/Q0 + /Q3*/Q2*/Q1*Q0 + /Q3*/Q2*/Q1*/Q0;

END GRAY1;
```

Figure 4. Sequential Design for Gray Code Counter; Logic Equations

Figure 4 gives the sequential circuit in the form of logic equation for the input to D flip-flops in terms of their outputs. This is actually the last step in standard sequential design and these equations can also be generated by providing a more algorithmic approach to the design as is shown in Figure 5. In Figure 5 the actual states and state transitions are defined. The system then generates a set of next state decoding equations once the type of flip flop has been set.

```
" GRAY8

PROCEDURE GRAY8 (INPUT clock, reset;
        OUTPUT q[4] CLOCKED_BY clock);

IF reset THEN
        q = 0;
ELSE
        STATE_MACHINE gray STATE_BITS q STATE_VALUES GRAY_CODE;
                STATE s1:
                        GOTO s2;
                STATE s2:
                        GOTO s3;
                STATE s3:
                        GOTO s4;
                STATE s4:
                        GOTO s5;
                STATE s5:
                        GOTO s6;
                STATE s6:
```

```
                GOTO s7;
        STATE s7:
                GOTO s8;
        STATE s8:
                GOTO s9;
        STATE s9:
                GOTO s10;
        STATE s10:
                GOTO s11;
        STATE s11:
                GOTO s12;
        STATE s12:
                GOTO s13;
        STATE s13:
                GOTO s14;
        STATE s14:
                GOTO s15;
        STATE s15:
                GOTO s16;
        STATE s16:
                GOTO s1;
    END gray;
END IF;

END GRAY8;
```

Figure 5. Sequential Design for Gray Code Counter; State Transitions

Conclusion

Using HDL in an introductory class has actual seem to help students to pick up some of the concepts more easily. In addition, it allows the students to approach interesting complex problems. The course has been taught using this approach for the past year and a half with favorable results. The students seem to accept and even like the material. This isn't too much of a surprise since most of the students today are very familiar with programming and using computers. For these students, the use of HDL seems quite natural.

References

1. John F. Wakerly, Digital Design Principles and Practices, Prentice-Hall, Englewood Cliffs, NJ, 2nd. Ed.,1994.
2. M.Morris Mano, Digital Design, Prentice-Hall, Englewood Cliffs, NJ, 2nd. Ed.,1991.
3. W.I. Fletcher, An Engineering Approach to Digital Design, 1980.
4. MicroSim Corporation, PLSyn Tutorial, 1995, 20 Fairbanks, Irvine, CA
5. Z. Navabi, VHDL, McGraw Hill, NY, NY, 1993.
6. J. Bhasker, A Guide to VHDL Syntax, PTR Prentice-Hall, Englewood Cliffs, NJ, 1995.
7. Allen Dewey, Analysis and Design of Digital Systems with VHDL, PWS Publishing, Boston, MA

Micheal E. Parten is an Associate Professor of Electrical Engineering at Texas Tech University. Dr. Parten has conducted research and published in the areas of education, instrumentation, control, modeling and simulation of a variety of systems, including semiconductor processing. Since returning to Texas Tech in 1984, Dr. Parten has served as the Director of the Undergraduate Laboratories in Electrical Engineering.