

VHDL Training Paradigms and Experiences

Tom A. Abels
Hughes Defense Communications
Fort Wayne, Indiana 46808
e-mail: taabel@most.fw.hac.com

ABSTRACT

The Hughes Defense Communications' Electronic Design Automation (EDA) support group has actively tracked and participated in the development of VHDL from its early days. Information on the language and the resultant impact on design methodologies was regularly disseminated to the ASIC/FPGA user community. Various training paradigms were used over the years to bring users to an effective mastery of VHDL. The training paradigm continues to change as we move to utilize our internal internet pages for interactive one-on-one training. This paper will review the paradigms used, the effectiveness and findings of each, and lay a course for future training experiences.

BACKGROUND

The established trend for the electronics industry demands that hardware be designed 33 percent faster and 33 percent cheaper than the year before. As a result, design tools and paradigms that were effective three years ago and sometimes only one year ago, may prove to be lacking in today's design environment.

In the research areas, both industry and university will lead the established design paradigm by several years. Thus the researched hardware description languages of the late 1970's and early 1980's, such as ZEUS [1], IDL [2], TI-HDL [3], and ISPS [4], predated the standardization and release of VHDL by as much as ten years. In the classroom however, universities frequently find themselves trailing leading edge design paradigms by several years. Even as late as two years ago, eight years after the 1987 release of VHDL, only 14 percent [5] of the graduating EE majors knew VHDL. Between these two extremes engineers are educated, receive their degrees, and are released into the work force.

Engineers in the work force that rely solely upon their university education will, in a matter of a few short years, find themselves woefully behind in the necessary design skills required by the ever-advancing, ever-demanding, design market. Engineers that desire to maintain a viable place in the electronic design market will expend considerable effort to read, study, and evaluate the ever-emerging design methods and tools. Those who refuse to do so will shortly find themselves "obsolete." This potential loss of a resource is something that industry cannot afford.

The electronic design industry depends heavily upon an educated, well-trained, work force. To maintain such a work force, industry is compelled to support this continuous demand to re-educate and re-train their engineers. Traditional instruction methods alone cannot accomplish this task, nor can the traditional providers of education unless they significantly expand the reach of their classrooms. [6]

At the same time, education is undergoing tremendous changes. There is less emphasis on rote memorization and repetition of facts. There is increased emphasis on problem solving, teamwork, and related thinking skills. Attention spans, heightened by a fast-paced MTV generation are shortened. We are saturated with information and information sources (e.g., perform a search on the World Wide Web). As a result, engineers and prospective students are forced to process and filter more information than ever before. This trend will only increase.

In this environment of upheaval in the worlds of design, education, and information, we need to train engineers to keep current. It is a formidable task.

THE EDUCATION PARADIGM

Adults in the work force are more diverse than either children or college students. There are great differences physiologically, psychologically, and sociologically. As such, educating adults will require a greater degree of flexibility within the classroom and the curriculum.

Adults, as a result of their increased years of experiences and previous educational paths will demand more not less in a training class targeted to them. Expectations must be clearly defined and stated. Student participation and discussion is a must. The psychological environment must be supportive, challenging, friendly, informal, open, and spontaneous without being threatening and condescending. [7]

It is paramount that the trainer is technically proficient in VHDL in order to teach the class. This technical proficiency will include the language, the underlying requirements and history of the language, the tools that utilize the language, and an extensive variety of applications that are relevant to VHDL. But technical expertise alone will not assure a successful training experience. In addition to technical expertise, the trainer must have a knowledge of the students and a knowledge of methods. [8]

In developing our VHDL training paradigm, the following five subjects were addressed:

1. Who to Train
2. When to Train
3. What are the Training Contents
4. How to Ensure Effective Training
5. Training Evaluation

Determining who is to be trained is frequently driven by project demand. However, once a core group of VHDL-literate ASIC/FPGA designers is established the focus turns from meeting project needs to developing a more educated and better skilled work force. With this change comes a definite need to refine and improve the training program. Students that took training due to a specific project need are less critical of the training and more willing to endure irritations and shortcomings. This is much the same situation found in EDA tool vendor training programs. Their audience, usually having just purchased a product, has a similar definite need for training. Once the prospective students are no longer those with a pending need for training but those who by

choice select the training, evaluations and criticisms will mandate that the class program be refined to make it more attractive and more robust. All aspects of the training including content, presentation material, handouts, location, class size, hours, etc., will need to be evaluated to provide the best training possible.

Ideally, the best time to train anyone on VHDL is several weeks or months before they will actually need to use it. And ideally, this lead time will likewise allow for free time and available tools in order to become more proficient with the language and the methodology before the assigned work begins. The ideal is not likely. Manpower assignments, project schedules, and work loads all reduce the likelihood of ensuring the ideal. Given that, determining when to train must allow for great flexibility. It is not uncommon that training will be required as part of the start-up of a project. In such cases there will have to be follow-up as the newly-trained engineers begin developing their first real VHDL designs. This becomes a critical point in helping them establish good coding styles and design techniques. From experience, this follow-up will frequently be of greater benefit than the original training.

In the VHDL training classes at Hughes Defense Communications we spend the greatest effort on methodologies and tool algorithms. In order to establish where VHDL fits within these methodologies, it is imperative that an understanding of the purposes and requirements of the language be thoroughly conveyed. If the requirements that VHDL be self-documenting and easy to read are well established, it will be easier to garner acceptance for the verbosity of the language. Understanding simulation algorithms and examining various synthesis algorithms will greatly enhance the designer's ability to select language constructs directed toward their end objectives.

Language syntax and tool usage skills are taught, but those subjects are not the primary focus. Down the road, designers will quickly search out missing or forgotten information regarding language syntax. Current tools will do much of this for them. They will not be as apt to revisit basic simulation algorithms for cycle-based simulation and how such an algorithm is impacted by various language constructs.

VHDL Tools, like EDA tools in general, are regularly being developed, refined, and released.

They are becoming more intuitive and easier to use. The work force, meanwhile, is becoming more computer savvy and is at the same time demanding more of the tools. The users do need to have an understanding of the capabilities and potentials of the tools to know what they can do with a tool. Current tools frequently come with on-line help, tutorials, and documentation. Users will be quick to utilize these aids to improve their tool usage skills in the areas applicable to them. The better they can do this outside of the class setting, the better they will be able to do it in the future as new tools and features emerge. However, they will be less apt to search out underlying strategies, methodologies, or algorithms that affect their design.

To ensure effective training requires closely matching the presented material, in both content and method, to the students. This requires preliminary knowledge regarding their experience and training background. It also requires an ability on the part of the trainer and the curriculum to be versatile. In addition, learning styles of the students must be understood. Some learn better visually, some auditorially, and some kinesthetically. Training methods must present the information in a way that all three are satisfied. [9] Utilizing a training method that focuses on a specific learning style may be necessary at times to pull in that student who is struggling.

Engineers enrolled in the class will do so for a variety of reasons. It is advantageous to know those reasons up front. Regardless, adults share some common motivations for extending their education. Adults need to be successful learners. Once the class has begun there must be experiences of success. If there are not, their motivation will be detrimentally affected. [10] To keep and maintain interest, effective questioning techniques should be employed. Questions should generally be composed ahead of time and should require individual and critical thought.

Questioning is described as the single most influential teaching act because of its ability to influence the learning process. [11] Associated directly with questioning is the art of discussion. This is the most favored and generally most effective teaching technique utilized with adult learners. [12] With VHDL classes, this is especially effective in areas where VHDL requirements are discussed. The class can readily find itself discussing some of the same pros and cons discussed in the original standardization efforts.

Discussion also works well in letting the class "discover" tool algorithms and coding strategies.

Vital to assuring and maintaining effective training is a thorough evaluation of the class. Each class and each offering should be evaluated. Special attention should be paid to nonverbal cues given throughout the class. The end-of-class evaluations should allow for anonymity and should have a section where the student evaluates their own performance and participation. Personal follow-up interviews can also be used to assess the class and get suggestions for improvement. From these evaluations, clear and specific tasks should be implemented to upgrade the training program.

A HISTORICAL PERSPECTIVE

Over the last 13 years we have held numerous classes to teach VHDL and given many presentations to share information on VHDL. Through those years both the content and the methodology has changed in order to better fit the audience and the growing maturity of the EDA industry.

Initially there were the VHDL Development Years (1984 to 1988). The language was going through various standardization efforts. EDA vendors were jockeying to carve out their niche in the expected HDL design market. At this point we began HDL training for ASIC designers. Classes were for three full days. The primary objective of the class was to convey the basic concepts of HDL-based design. An overview of the purposes and the requirements of VHDL were stressed. This would later prove to be very advantageous as those engineers who had the best understanding of these basics went on to utilize the language more proficiently. Later they also became the quickest to pick up on higher levels of abstraction in their design work. In these initial classes syntax for vendor-specific languages were used (e.g., Verilog from Gateway Design Automation and DABL from Daisy). Students were encouraged to use language constructs which were similar to developing VHDL constructs. Minimal instruction time was given to actual tool-usage skills. Initially this minimal focus on tools was because the vendor-specific languages were considered temporary. Their planned demise would coincide with the standardization of VHDL and the release of VHDL tools. This initial training paradigm proved to lay a sound foundation and provided some of our best

VHDL designers. The basics of this paradigm are still employed.

Then came the Early Years of VHDL (1989 - 1991). VHDL was officially standardized and EDA vendors began delivering VHDL tools. The syntax of VHDL could finally be taught in earnest. But the foundation for training continued to be the basics of HDL design and the overview of the purposes and requirements of VHDL. EDA vendors were employed to teach their tool and VHDL. Four vendors who claimed to provide a simulator that supported the whole of VHDL-1076 were brought in and each taught a class to a team of five engineers. Both the tools and the classes were evaluated. Instruction time given over to the operation of VHDL tools was primarily left in the hands of the vendor-provided trainers. This training paradigm proved to be both less effective and more expensive than any paradigm that preceded or followed. The foundational HDL/VHDL information that from the initial training paradigm proved to be quite effective was not adequately provided by outside EDA trainers. Likewise, outside trainers tended to focus on VHDL constructs that were handled well or enhanced by their specific tools. Constructs that would merge well with the varied backgrounds of the students, that would lend themselves to uniform coding styles, or that would facilitate code reuse were not a primary emphasis. Engineers that were trained in this period had the hardest time jumping from previous design methods to HDL-based design methods. Most of students opted for subsequent training when it became available.

Shortly thereafter, we found ourselves in the VHDL Growth Years (1992 - 1994). Usage was growing at a dramatic rate, both within our own design groups and in the ASIC/FPGA design industry as a whole. Outside EDA-vendor supplied trainers were no longer utilized due to both the cost and their shortcomings. An internally-developed week long class was structured to provide 20 hours of class and 20 hours of lab work on VHDL tools. This training continued to adhere to the same foundation of basic language-based design concepts and the underlying purposes of the VHDL language. In addition, heavy emphasis was now being placed on design methodology and software programming techniques. Very little class time was spent on VHDL tool skills. Training in those areas came from detailed lab instructions that relied upon vendor-supplied documentation and training aids. This training paradigm proved to be

an efficient way to train capable VHDL designers. It also provided overload to the students. A great amount of information was being conveyed to each student in a short amount of time. The labs required at least a rudimentary level of understanding of the material and there was not a lot of time to digest it well. To accomplish this in five days was quite taxing to the student. Engineers trained with this paradigm did well, but were more apt to return to their training materials weeks and months later to review or learn material that had never really been absorbed.

Currently, we are in the Years of the Upward Extension of VHDL (1995 - 1996). VHDL is being extended to more abstract design and system design. Synthesis is an integral part of language-based design. These changes initiated a similar change in our training model. Presently, our VHDL training class lasts for 10 weeks with four hours of class per week. Labs, for the first time, were done outside of class. Class material was expanded to include additional methodology and synthesis curriculum. This paradigm reduced the feelings of "information overload" experienced by the students, a primary complaint from the previous paradigm. The length of the class has allowed for more reading assignments, more complex labs, more time to absorb the material, and time to handle make-up work. The more complex labs allowed for a class project that required the combined effort of the class, discussion and analysis efforts, and could be directly tied to design work already scheduled for students. However, the primary objections to this paradigm are the length of time and the single pace to which the class is subjected.

The Future Years of VHDL (1997 - ?) will also demand changes in the training model. At this time we are prototyping and evaluating a new paradigm. In addition to improving the training experience, our added objectives are: To reduce training costs, incorporate new material, and allow the student to learn at their own pace. The new material includes additional synthesis material, programming standards, reuse guidelines, and system design techniques. To this end, training material is being ported to our internal internet Web pages which can be accessed through the VHDL VUG (Virtual User Group) also located on our intranet Web pages. Trainer time will be required when questions arise regarding lab work or class exercises, but it is expected that most of this support can be handled via phone and e-mail. This paradigm will allow for training to be

accessible to all personnel. It will also allow for them to progress through the training material at their own pace.

ANALYSIS

Each training paradigm has its strengths and weaknesses. In retrospect, the greatest benefits have been a direct result of the degree to which design methodologies and tool algorithms were discussed. Some of this same material can and should regularly be presented to project and program leaders in order to keep them abreast of the direction ASIC/FPGA and system design is moving. It is not a static world and frequently they will find their engineers working with design methodologies and languages and tools that they find quite unfamiliar. This can lead to significant conflicts.

First time students will frequently approach VHDL training with fear, apprehension, and even antagonism. Two attitudes conveyed by the instructor will help alleviate some of this. The first is to treat the students with a normal positive expectation that they will learn. The second is to make the training worthy of the student's choice to participate in it. [12]

Lab work, especially class projects, where time is required to analyze the problem and evaluate various algorithms and levels of modeling, is vitally important. Adult students have a strong need to apply what they have learned and to be competent in that application. [13] This is also why it is advantageous to encourage students upon completion of class to get experience using the VHDL skills they have learned in class.

Each student, before class begins, should fill out a survey detailing their education and experience in several related fields, EDA, HDL's, Software Development, System design, UNIX, and ASIC/FPGA design. This survey should also yield an understanding of the student's potential uses for VHDL and to understand their areas of design expertise. This allows for class examples and discussions to be tailored to better include all students in the class.

As a primary purpose for VHDL training is to have a better educated work force, we have no stipulated prerequisites for students. However, it is conveyed that those lacking a working knowledge of UNIX, those having little or no software development background, and those with

no EDA experience, will have some additional learning in order to progress along with the rest of the class. Depending upon their schedule this may be done before class or in conjunction with the class.

The most successful students are those who are willing to undertake a challenge. Both the language and the associated methodologies will present a challenge. Learning new design languages and methodologies requires a level of critical thinking that doesn't come without a desire to question assumptions underlying the traditional way of design. These students will be the ones to question the existing design paradigm as tomorrow's begins to take shape.

FOLLOW-UP

Training is a starting point. It is not the end. Proficiency will come with continued use. With this use will come personally developed styles and methods. In order to maintain a level of consistency across designs it is imperative that changing methods and styles required by the employer or the program be communicated back to the designers.

Coding styles and design methodology is an art, an orderly approach to the task at hand, and to be effective the methodology must be built upon a clearly defined set of rules and requirements. [14]

An avenue by which updated information can be disseminated to formerly trained VHDL designers should be clearly defined and presented to them when they are trained in order for them to look for and plan for updates. Ideally this information will be accessible on-line by any designer. User groups work well in providing this capability. However, getting consistent, wide-area attendance in a user group from designers that have varied interests, project demands, and time limitations is a difficult task. We have opted to create "Virtual User Groups" (VUGs).

These VUGs assist us in providing our follow-up support. Critical information on tools, versions, installations, changing methodologies and technologies are all presented on these Intranet pages. Access to design data archives, internal papers, and style requirements are also available. Additions and updates may be submitted by any user with access to the system. Previously trained designers are notified when significant updates are

made to the VHDL VUG. It is on these same pages that we are placing all current VHDL training materials, examples, tutorials, labs, and supporting papers. Links within these pages allow for access to a wealth of information on VHDL related topics that exist on the Internet. As this VUG matures it will allow for stylized training in addition to excellent follow-up potentials.

With our fast-paced ever-changing design environment, it is imperative that the training effort be carefully planned and integrated with the existing EDA follow-up and support efforts found within the organization. Unfortunately the topic and issues related to training, like this paper, are frequently found to come last.

- [1] Lieberherr, K. J. and Knudsen, S.E., "ZEUS: A Hardware Description Language for VLSI," 1983 ACM IEEE 20th Design Automation Conference Proceedings, June, 1983.
- [2] Maissel, L. I., and Ostapko, D. L., "Interactive Design Language: A Unified Approach to Hardware Simulation, Synthesis, and Documentation", ACM IEEE 19th Design Automation Conference Proceedings, June, 1982.
- [3] "VHSIC Phase 1 Program HDL/Design Simulator User's Manual," Texas Instruments, Dallas, Texas, Nov. 1983.
- [4] Barbacci, "Instruction Set Processor Specifications (ISPS): The Notation and Its Applications," IEEE Trans. Computers, Vol. C-30, No. 1, Jan. 1981.
- [5] Bellinger, Robert, "Verilog Training Lacking," Electronic Engineering Times, June 12, 1995.
- [6] Eurich, Nell P., "The Learning Industry: Education for Adult Workers," The Carnegie Foundation for the Advancement of Teaching, Princeton, NJ, 1990
- [7] Knox, A. B., "Helping Adults Learn: A Guide to Planning, Implementing, and Conducting Programs," Jossey-Bass, San Francisco, CA, 1986
- [8] Knox, A. B., (Ed.). "Teaching Adults Effectively," New Directions For Continuing Education, No. 6. Jossey-Bass, San Francisco, CA, 1980
- [9] Laborde, Genie Z., "Influencing With Integrity: Management Skills for Communication and Negotiation," Syntony Publishing, Palo Alto, CA, 1983
- [10] Spence, J. T., "Achievement and Achievement Motivates", Freeman, San Francisco, CA, 1983
- [11] Taba, H., "Teaching Strategies and Cognitive Functioning in Elementary School Children," San Francisco State College, San Francisco, CA, 1988
- [12] Galbraith, Michael W. (Ed.), "Adult Learning Methods," Kreiger Publishing, Malabar, FL, 1991
- [13] Knox, A. B., "Adult Development and Learning: A Handbook on Individual Growth and Competence in the Adult Years," Jossey-Bass, San Francisco, CA, 1977
- [14] Cohen, Ben, "VHDL Coding Styles and Methodologies ... an In-Depth Tutorial," Kluwer Academic Publishers, Norwell, MA, 1995