Microelectronic circuit test engineering laboratories with programmable logic

Ian Grout and Joseph Walsh
Department of Electronic and Computer Engineering, University of Limerick, Limerick, Ireland
E-mail: Ian.Grout@ul.ie

Abstract A use for programmable logic is presented targeting microelectronic circuit test engineering education. These devices offer strengths in cost, time and flexibility in an educational environment. The rationale for this work is to support electronic hardware design, fabrication and test for mixed-signal integrated circuits typically used in mixed-technology systems.

Keywords microelectronic circuit test; programmable logic

Today, the use of the programmable logic device (PLD) is pervasive in educational and industrial applications. The ability to configure and readily modify complex digital circuits and systems provides many time and cost advantages. This paper will discuss a use for PLDs[1,2] within an electronic engineering educational environment, along with the potential benefits that the use of these devices can provide to the student learning experience. Specifically, the development of laboratory experimentation based on the use of complex PLDs (CPLDs) will be discussed. Here, the Lattice Semiconductor Corporation MACH-Series CPLD[3] is primarily considered. With the rapid increase in the use of electronic and computer engineering solutions within a mixed-technology (the *mechatronic*) environment, the need for 3rd level (university) graduates with the right mix of electronic circuit/system design, fabrication/manufacture and test skills, is even more essential than ever before. In this paper, an electronic engineering perspective is taken and the need primarily for electronic circuit/system test engineering education is discussed. The use of digital programmable logic is an integral part of this, with a *Microelectronic Test Engineering Education Development Platform* design and development, based on CPLD technology, introduced. The rationale for this is to allow the introduction of test engineering specific concepts and implementation issues within an undergraduate level laboratory teaching environment. This is to be seen as an integral support role within the development of complex mixed-technology systems.

The experiments provided are aimed at undergraduate (and potentially) postgraduate students, within a 3rd level teaching and learning environment, undertaking introductory modules in microelectronic circuit test engineering.[4–6] In the main, digital integrated circuit (IC) test engineering concepts are introduced[7,8] in taught modules with a strong practical emphasis and content. This allows for the use of the digital programmable logic devices currently available. However, the need to consider and introduce the concepts for mixed-signal IC[9] test is also an integral part of the developed system. In particular, data converter (analogue to digital (A/D) and

digital to analogue (D/A)) device test is supported. The rationale for this approach is considered as follows:

- Within a national economic context, microelectronic circuit test is becoming an increasingly important area of activity for the microelectronics industry based in Ireland. The move over the last few years has been away from industrial *back-end* production test activities and more towards a *front-end* test strategy and development role. This is leading to the need for the development of new engineer and technologist skills that needs to be a concern of the university sector.
- Today, in many cases, the requirements placed on the test engineer within the electronics and microelectronics industries will have little resemblance to the requirements in years gone by. This in many cases is due to the substantial increase in size, functionality, and complexity of the product that is to be tested, along with the introduction of new fabrication/manufacturing processes.
- An engineer involved in both integrated circuit test development and production test activities will be required to be involved in a range of tasks that are performed from initial IC design concept through to final product delivery to the customer.
- High performance analogue, digital and increasingly mixed-signal ICs are encountered, from analogue (operational) amplifiers through data converters to complex, high-speed digital signal processors with analogue & digital I/O.
- *Design*, *Test* and *Fabrication* are interleaved processes, with complex interactions at various stages of a product development process. Test, once seen as an afterthought, is now a proactive activity starting at a design concept stage. The move is towards an integrated *Design for Testability* (DfT) approach for electronic and microelectronic circuits and systems, see Fig. 1.
- Test, above many other activities with the microelectronics field, requires a sound practical knowledge of concepts and design requirements, given practical limitations of the devices to test, the development of test methods and production software test programs, along with the limitations of the test equipment to be utilised.

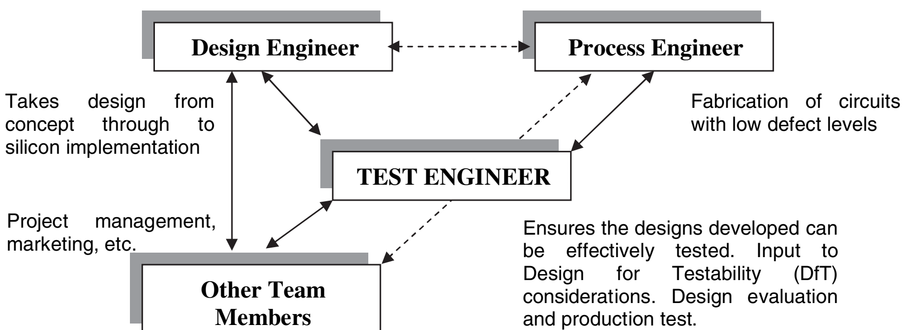

Fig. 1 *Test engineering, seen as the link for successful product development.*

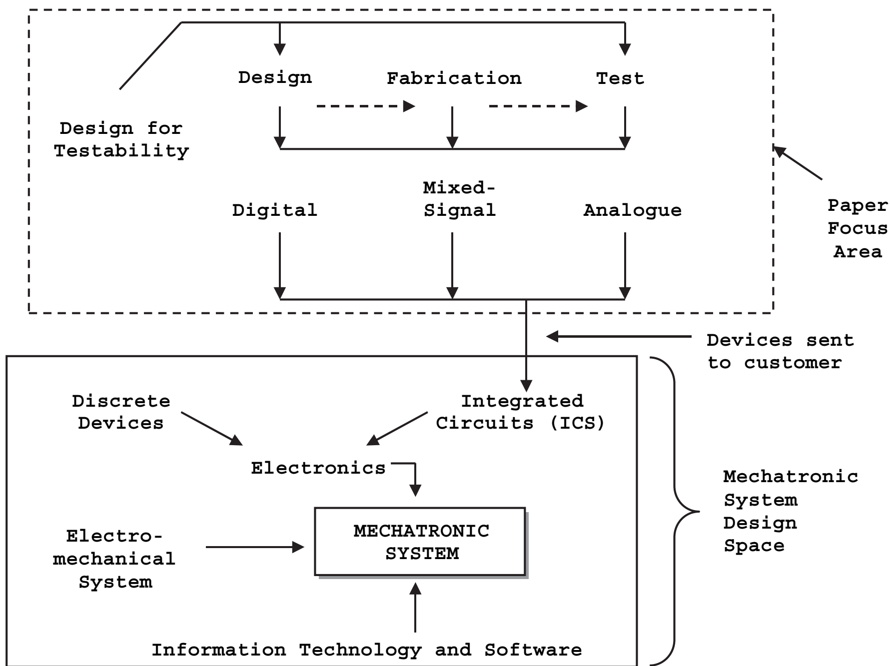

Fig. 2 *Focus area of paper.*

The educational programmes delivered to the student are required to take into account these considerations and to migrate educational programmes towards the delivery of new course material with the right levels of quality and relevance. The purpose of this paper is to discuss the introduction of test engineering concepts aimed to support the introduction of a relevant graduate skills base. The focus of the paper is represented in Fig. 2.

The context for the use of final electronic/microelectronic circuits and systems would typically be in a mixed-technology (mechatronic) target system.[10–12] Whilst the designer of the non-electronic parts of a mechatronic system may not necessarily have any involvement in the steps involved in an IC production (from concept through to delivery to the customer), the importance of test and the necessity to identify and implement effective test procedures cannot be underestimated. The electronic circuit/system designer would ideally have a good knowledge of test as a complementary skill to design.

Electronics within a mechatronic environment

The electronics that are incorporated into a mechatronic system are required to undertake a range of activities from sensor interfacing, data sampling and analysis through to actuator stimulation. The need for high performance, high quality elec-

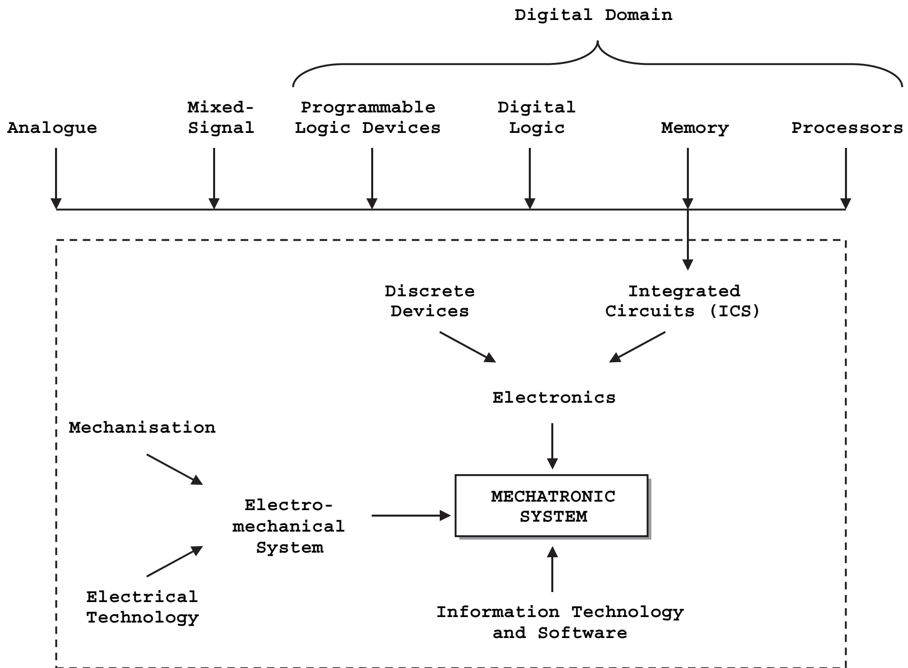

Fig. 3 *Electronics in the mechatronic environment.*

tronics, available to the designer is essential to the success of a design project, along with the choice of the right technologies to implement the required functions. The choice as to which type of electronic circuit/system implementation to utilise is non-trivial. Figure 3 highlights the types of electronic devices that would be used. Digital, analogue and mixed-signal (in the main data converter (ADC and DAC) designs would typically be required). This figure also shows the key constituents of a mechatronic system:

- *Electro-mechanical system*: Mechanical and electrical parts to form mechanical actuation via electrical principles (e.g. motors).
- *Information technology and software*.
- *Electronics*: considering in more detail, the electronics being either discrete components (e.g. passive devices (resistors, capacitors and inductors), discrete semiconductor devices (diodes, transistors, power semiconductors), or as integrated circuits (ICs)).

With the increase in utilisation of embedded systems and the concept of 'embedded intelligence', the size and complexity of the electronics is set to increase dramatically. Additionally, a required circuit functionality could be implemented in a range of means. For example, taking a closed-loop control algorithm. This could be implemented in the analogue domain using operational amplifier-based circuitry (for

example, a PID (proportional plus integral plus derivative) algorithm). However, this could be implemented digitally using a processor (microprocessor, microcontroller or digital signal processor (DSP)) running a software program, in hardware using discrete logic ICs or with programmable logic. In programmable logic, the choice in many cases is realistically based on suitable FPGA (field programmable gate array) or CPLD technology. However, in many cases a software programmable processor would be the desired choice. The ability to modify operation via software re-coding whilst maintaining the same electronics (no electronic circuit re-design time and costs required, only software) is a desirable feature, either for in-system calibration, system tuning, ease of future upgrading, or simply where the design is created without full knowledge of the final system dynamics and the need for flexibility exists.

Programmable logic technology and utilisation

In this section, programmable logic[1–3] will be considered as the digital hardware devices that may be configured to form a range of digital circuits and systems. Such devices also provide a good example of how over the last number of years, the line between hardware and software has blurred. With electronic hardware design engineers creating the bulk of their new digital circuitry in programming languages such as VHDL[13] and Verilog-HDL[14] and often target it to programmable logic technologies, this is changing the way electronic systems are designed. In the simplest terms, a single PLD can replace a PCB containing multiple COTS (commercial off the shelf) discrete digital ICs, providing a potentially smaller and faster solution with lower power consumption. However, these devices have more use with their ability to rapidly prototype complex ASIC (application specific integrated circuit) designs through to in-system upgrading via a local programmer or remotely via the Internet. Reconfigurable devices can be identified as shown in Table 1. This also includes the move towards providing reconfigurable analogue and mixed-signal functionality within a programmable device.

The current range of programmable logic on offer includes everything from small devices capable of implementing only a handful of logic equations to large FPGA

TABLE 1 *Programmable device types*

Domain	Type		
Digital	PLD (programmable logic device) FPGA (field programmable gate array)	SPLD (simple PLD) CPLD (complex PLD)	PLA (programmable logic array) PAL (programmable array of logic) GAL (generic array of logic)
Analogue and mixed-signal	FPAA (field programmable analogue array)		

TABLE 2 *Main programmable logic vendors and device summary*

Vendor	Internet resource
Altera Corporation	http://www.altera.com
Xilinx Inc.	http://www.xilinx.com/
Lattice Semiconductor Corporation	http://www.latticesemi.com/
Cypress Semiconductor	http://www.cypress.com
Atmel Corporation	http://www.atmel.com

'*Systems*' that can implement a complete processor system, including peripherals. A brief list of the key vendors is shown in Table 2.

With ASIC designs, the time required to go from design, through prototyping, to final manufacturing, can take from several months to more than a year, depending on the complexity of the device. And, if the device does not work properly, or if the requirements change, a new design must be developed. The up-front work of designing and verifying ASIC devices involves substantial non-recurring engineering (NRE) costs. These represent all the costs incurred before the final device emerges from a silicon foundry, including engineering resources, software design tools, photolithography mask sets for manufacturing the various layers of the chip, and the cost of initial prototype devices. With programmable logic devices, designers use suitable software tools to quickly develop, simulate, and test their designs. Then, a design can be quickly programmed into a device, and immediately tested in a live circuit. The PLD that is used for this prototyping is the exact same PLD that will be used in the final production of a piece of end equipment, such as a network router, a DSL modem, a DVD player, or an automotive navigation system. There are none of the NRE costs associated with an ASIC design, and the final design is completed much faster than that of a custom, fixed logic device. Another key benefit of using PLDs is that during the design phase, users can change the circuitry as often as they wish until the design operates to their satisfaction – in order to change the design, the device is simply reprogrammed. Once the design is complete, customers can go into immediate production by simply programming as many PLDs as they need with the final software design file.

Increasingly, the programmable devices used are based on CPLD and FPGA technology. These devices are used in industrial and educational environments since they can offer reconfigurability and rapid prototyping benefits. Specific types of device can be either 'configure once' without the opportunity of modifying the design after configuration (anti-fuse based), or reconfigurable (reprogrammable memory based configuration). Figure 4 shows the two methods to program the reconfigurable devices, the method dependent on the device type and end-use environment.

Usually the configuration (and verification) operations via the PC are based on using the JTAG (Joint Test Action Group) Boundary Scan standard (IEEE standard 1149.1),[15] a design for testability (DfT) technique used for digital systems test.

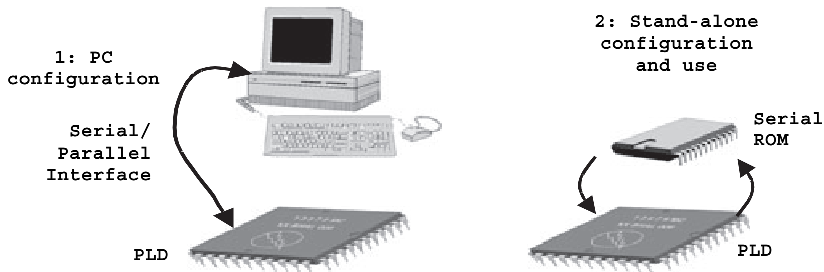

Fig. 4 *Configuration methods.*

The need for microelectronic circuit test engineering education

The increasing design complexity and reduced error margins in semiconductor manufacturing are forcing design and test engineers within industry to be ever diversifying in the range of activities undertaken on a day-to-day basis. However, in the test side, no matter how complex a microelectronic circuit design becomes, testing of the design relies on basic concepts, which can be obtained from suitably defined 3rd level education.[16] There are a number of avenues that can be followed in the area of microelectronic test education, depending on the skills set requirements of the students: consideration must be given to whether it is hardware or software test that is the primary concern. In this work, electronic hardware test is the primary concern. In hardware test, the main areas of concern are device (IC) level or systems level. Today, however, it should be noted that the boundary between device and system is less well defined with the advent of the '*system on a chip*' (SoC), where the concern is to integrate as much of an electronic system as possible onto the silicon die.

The teaching of test engineering at the university level is critical to providing the students with the necessary skills to graduate into a range of industry-based engineering positions, primarily those who will be focused towards associated test engineering activities, but also those who may be involved in activities from design through to production. Dedicated test engineering taught modules within a degree scheme of study need to address both device level test, system level test and production test systems. The correct amount of local and possibly remote learning[17–19] also needs to be addressed. Table 3 lists the requirements on the educational side of each of the test engineering areas mentioned, considered in the context of the teaching and learning needs identified in the paper.

Test engineering development platform

The practical prototyping phase within design and test modules has in general been very time consuming, and has resulted in hardwired implementations that are frequency limited and prone to errors during the circuit build. This can result in the laboratories being of limited benefit, when the underlying test engineering princi-

TABLE 3 *Key aims of test engineering modules*

Device level test	System level test and production test systems
• Provide a discussion into, and application of, test engineering concepts in the testing of integrated circuits during, and post-fabrication. • Introduce the design and testing of specific circuit architectures commonly encountered in digital and (to a limited extent) mixed-signal ICs. • Introduce and discuss *Design for Testability* (DfT). • Practise the *build & test* of an electronic circuit emulating a complex digital logic IC in both fault-free and under specific fault conditions (the circuit under test (CUT)) within the structured laboratory sessions.	• Provide a discussion into, and application of, test engineering concepts in the testing of electronic systems (PCB, SoC and MCM based) and the automatic test equipment (ATE) requirements. • Discuss system level test issues, the causes and effects of system faults. • Discuss system level test, in particular the structure and use of the IEEE 1149.1 (Digital Boundary Scan) & 1149.4 (Mixed-Signal Test Bus) standards and implementation. • Practice fault simulation and DfT insertion through the design entry and simulation of digital logic designs.

ples that were to be introduced are masked by problems in the circuit build stage of the laboratory class. One reason for this is that the final circuit implementations usually incorporate a large number of low density discrete (small scale integration (SSI) and medium scale integration (MSI)) ICs which are interconnected together in order to show proof of concept by using prototyping board or wire-wrapping techniques.

As an alternative, a laboratory environment that would allow a user to rapidly implement a number of application circuits using a suitable CPLD and ECAD (electronic computer aided design) software tool for design entry, to rapidly download the circuit configuration to the CPLD with the aim to test and evaluate the operation of the design, may be of great advantage to students. This would enable the reduction, or elimination, of the circuit wiring stage unless this was considered to be part of the learning experience. This type of development environment would allow students, to configure a range of designs (e.g. in digital, combinational logic circuits, sequential logic (state machines and counters), memory blocks, registers, arithmetic and logic unit (ALU) and simple CPU type of functions) onto a single device, and develop suitable test vectors for either functional or structural testing of digital logic (and with extensions to the above, mixed-signal) designs.

The students could, in addition to the implementation and analysis of basic designs, implement DfT concepts such as built in self test (BIST).

Built in self test (BIST)

To improve the testability of a design, a structured design for testability (DfT) approach should be adopted. Here, the design is generated to enable the right level of circuit accessibility (controllability and observability) by the inclusion of test specific circuitry. A DfT approach will require additional circuitry in order to select the

design between the normal operating mode and one or more test modes. DfT can aid the *traditional* test approach whereby an external tester is utilised to apply test patterns and monitor the circuit response. The external tester will then compare the actual device response to a fault free response stored within the tester. From this, it can be ascertained whether the device passes or fails the test. Increasingly, designs are being developed which incorporate BIST. Here, the design itself is responsible for generating test vectors, applying these vectors and monitoring the response in order to ascertain whether the part of the design tested using the BIST passes or fails the test. The main advantages of BIST are shown in Table 4.

Test patterns for BIST can be generated at-speed using a linear feedback shift register (LFSR) with only a clock input. A LSFR is a fast counter that requires limited resources. It counts in a pseudorandom sequence that repeats after a maximum of ($2^N - 1$) cycles, where N is the number of bistables in the circuit. A LFSR is the heart of any digital system that relies on pseudorandom bit sequences (PRBS), with applications ranging from cryptography, bit-error-rate measurements, wireless communication systems and microelectronic test applications. The outputs of the circuit under test (CUT) must be compared to the known good response. In general, collecting each output response and off-loading it from the CUT to the tester for comparison is too inefficient to be practical. The general solution is to compress the entire output stream into a single signature value. An example of a basic BIST scheme using the LFSR to implement an on-chip signal generator and analyser in order for self-test of a CUT, is shown in Fig. 5. A LFSR case study example is discussed in detail in the next section.

The designs mentioned could also be integrated into larger systems, allowing

TABLE 4 *The main advantages of built in self test (BIST)*

1	Eliminates the need to generate and apply a large set of test vectors externally via the I/O pins.
2	Permits testing at speed under real-world conditions.
3	Improves access to internal nodes.
4	Ensures that test issues are addressed early in the development cycle that results in higher quality, first-time designs.

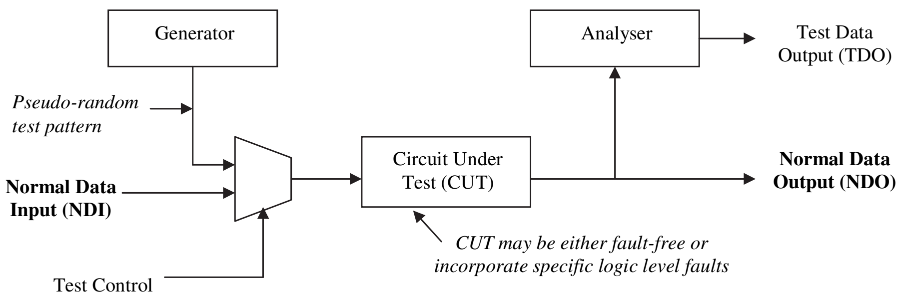

Fig. 5 *Built in self test (BIST) arrangement.*

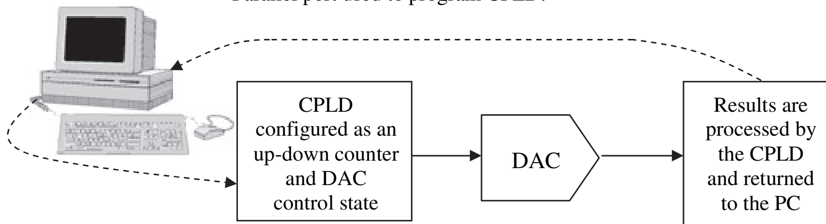

Fig. 6 *Example mixed–signal test arrangement.*

individual blocks to be partitioned and tested in order to model industrially relevant problems. The set-up can also allow mixed-signal test by utilising the on-board ADC and DAC devices and interfacing these devices to the programmable logic as shown in Fig. 6. Here, the type of experiment would be to undertake ADC and DAC test. From the circuit outputs the students could determine pertinent parameters such as INL (integral non-linearity) and DNL (differential non-linearity) with a mathematical analysis toolset such as Matlab.

When developing the learning environment, (i) flexibility and (ii) ease of use were the two key requirements. In this case, the Lattice Semiconductor MACH4 CPLD was chosen. While it was possible to utilise a range of programmable logic technologies, such as the Xilinx family of devices, or larger CPLD devices, this particular device was initially chosen for the follow reasons: – low entry cost and availability of the complete development hardware. The basic system consists of a PC bsed tester environment where the user starts a suitable interface program written in a language such as Visual Basic (VB), and the Lattice Semiconductor software toolset.[17] Both tools then interface through the PC serial port (RS232) for circuit run-time data I/O and the PC parallel port for device programming. When the user is ready, a new or a previously designed circuit is downloaded on to the CPLD, which is located within the ULTEDB (University of Limerick Test Engineering Development Box). The user can produce a test vector set for the circuit using the Graphical User Interface (GUI) in order to determine suitable test vectors for the particular design. This system can provide a new fast and effective way of teaching microelectronic test engineering concepts. The intention of this complete microelectronic test engineering environment, as shown in Fig. 7, is to allow students to design basic circuit building blocks and architectures, and from this introduce DfT concepts.

An advantage of this approach is that alongside the ability to rapidly prototype circuits, is the ability to introduce faults into the circuit and to allow the students to discover whether a particular circuit fault can be detected. However, it may be that the circuits provided do not actually include a fault, or the faults maybe undetectable. The potential for this type of system is not limited to microelectronic test engineering modules. The system is suitably generic in that it could be of use in courses such as digital logic design, VLSI design, introduction to hardware description languages

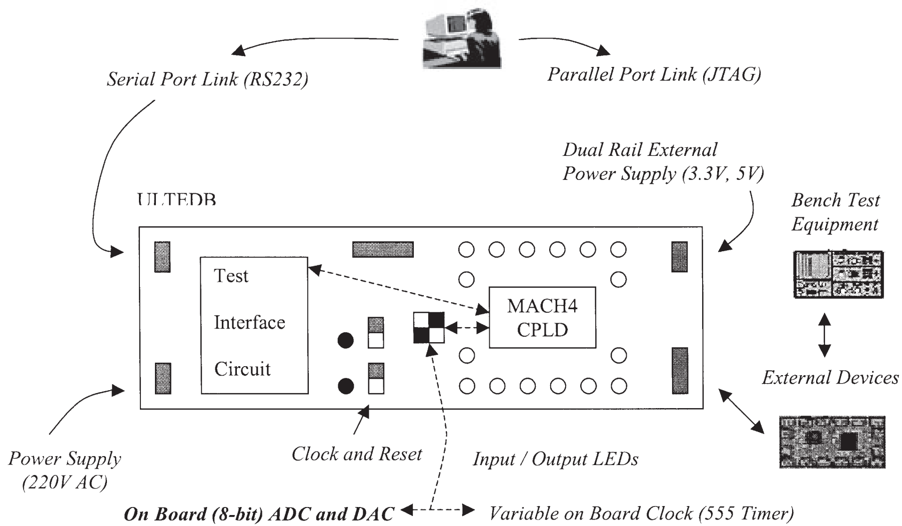

Fig. 7 *ULTEDB environment.*

(HDLs), computer science, computer architecture, mechatronics, etc. The ability to communicate easily with external devices, such as ADCs and DACs from the PC also opens the door for analogue electronic and optical test. Additionally, building up a library of circuits and case studies will allow students to develop better microelectronic test engineering skills on a year by year basis, which in time can help improve the overall quality of microelectronic test engineering. A photograph of the complete laboratory set-up is shown in Fig. 8.

In particular, the ULTEDB allows for the user to insert logic fault models on a single or multiply fault assumption as shown in Table 5.

ULTEDB laboratory case study

A case study laboratory was set-up and used by the test engineering students in the last academic year. The aim of this laboratory session was to investigate the operation and testing procedure of a linear feedback shift register (LFSR) circuit using the ULTEDB development platform. The CUT used is shown in Fig. 9. The CUT was to be considered as a single complex IC where access to control and monitor of the circuit operation was via the primary inputs and outputs only, as shown in Fig. 10.

The experiment was laid out to investigate design and test issues relating to a LFSR circuit, in particular circuit design issues which may result in certain faults being undetectable. The ULTEDB CPLD contained the LFSR circuit. The students were instructed to follow the laboratory manual and to investigate as to whether a SAF existed in the circuit or not. The students used the ULTEDB development

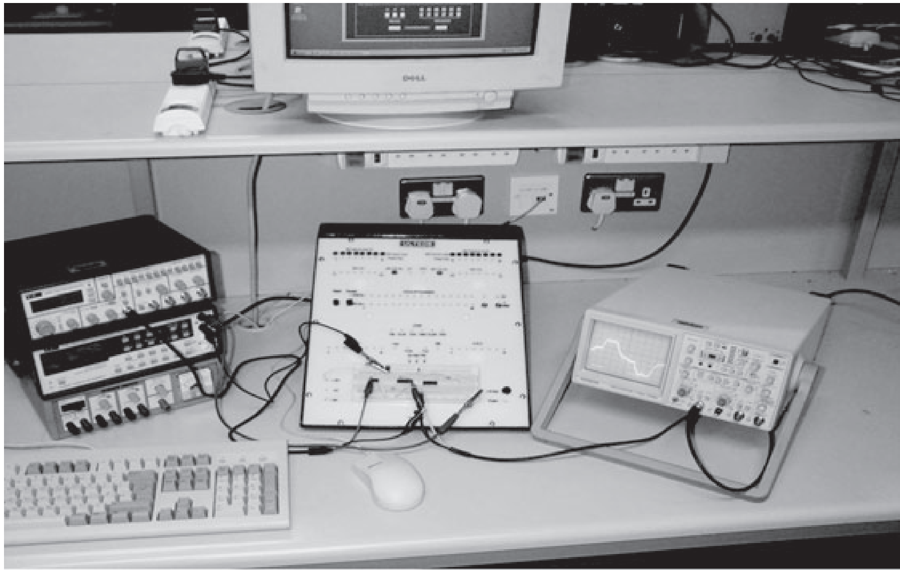

Fig. 8 *ULTEDB laboratory environment.*

TABLE 5 *Logical fault models considered*

Single digital stuck-at-fault (SAF)	*The SAF model is the most popular fault model. The single SAF makes the assumption that only one line in the gate or cell is faulty at one time and that the fault is permanent as opposed to transient.*
Bridging fault (BF)	*The bridging fault model assumes that two nodes of a circuit are shorted together. This failure mode becomes more important as IC line widths and pitch get smaller and their aspect ratio increases. Logical wired-AND and wired-OR bridging faults are considered.*
Multiple stuck-at fault model (MSAF)	*The multiple stuck-at-fault (MSAF) models makes the same basic assumptions as SAF, except it allows two or more lines in the circuit to be faulty at the same time.*

platform, and a specifically designed Visual Basic user interface to apply a series of test patterns to reset and clock the LSFR to each state and record the state truth table. This was then compared to the theoretical state table that was also calculated, and identified the possible fault or faults in the circuit.

Potential benefits to the student learning experience

The purpose of the experimentation platform utilising programmable logic provides a great deal of potential flexibility in the use of the same platform for a range of

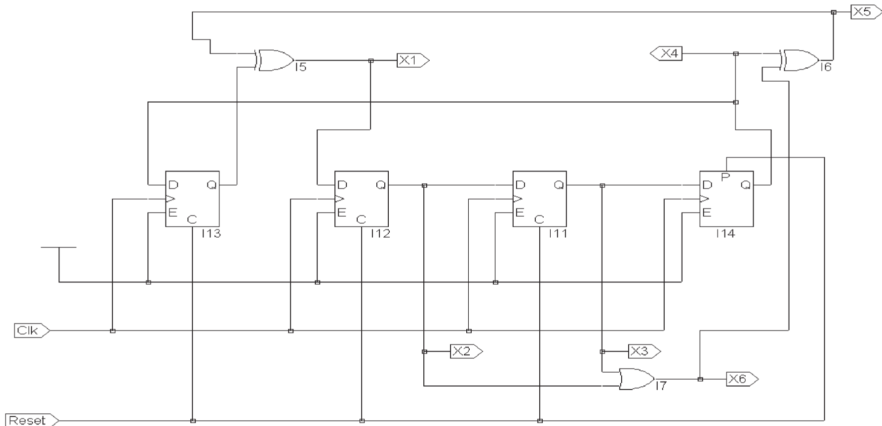

Fig. 9 *Linear feedback shift register (LFSR).*

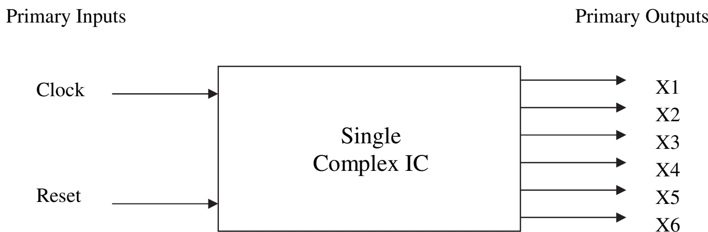

Fig. 10 *Single complex IC.*

teaching and learning scenarios. This can provide for a cost-effective solution to the need to provide hardware and software teaching equipment. Table 6 summarises the scenarios that have been considered. The scenarios are seen from three perspectives:

(i) *Semiconductor test program development*: the need to develop cost effective and high quality test programs to detect faults (process variation and hard defect related).
(ii) *Digital logic design techniques*: designing the types of circuits and systems required in today's operating environments.
(iii) *Mixed-technology systems design*: the design of systems where the electronics forms a key part of the system operation.

Whilst the unit has been developed primarily for final-year undergraduate students undertaking education programmes in semiconductor test, the flexibility provided by the unit in terms of the ability to configure a range of circuits, lends the unit to a more flexible use in other electronic and electro-mechanical learning environments where digital logic forms an important part of the overall system to be developed and evaluated.

TABLE 6 *Teaching and learning scenarios*

Scenario	Details
	Microelectronic circuit test program development
1	Development of test patterns and applications to a range of combinational and sequential logic circuits. The ability to configure fault-free and faulty circuit behaviour (the actual status of the circuit being unknown to the student and the aim being to enable the student to determine whether a fault exists and can be detected, whether no fault can be detected (either a fault does not exist or a fault may exist but cannot be detected either with the patterns entered or at all)).
2	Learning the structure and use of the 1149.1 (JTAG) Boundary Scan standard via the programming and verification of the CPLD.
3	Development of mixed-signal test programs (in particular for lower resolution (8-bit) data converter designs).
4	Development of tester platform architectures. With the arrangement, a (modest) emulation of an ATE platform may be created for digital and mixed-signal IC test.
	Digital logic design techniques
5	Design and verification of combinational and sequential logic circuits.
6	Use of CPLD (and FPGA) technology, schematic capture, development and synthesis of hardware description language code, simulation and device programming.
	Mixed-technology systems design
7	Design and prototyping of electronics within a mechatronic system environment. For example, sensor and actuator interfacing, closed loop control, digital signal processing (DSP), PC-based control.

Conclusions

This paper has discussed the use of programmable logic in a teaching and learning environment. The target area was semiconductor test engineering with an emphasis on digital test concepts, along with an introduction to mixed-signal test. A laboratory experimentation unit based on CPLD technology that was developed at the University of Limerick was described and its uses outlined. Whilst aimed at semiconductor test engineering teaching, its potential use in other realms of engineering was highlighted. The need to consider semiconductor test education in its own right, alongside and with equal importance to electronic circuit design was discussed. The rising complexities and functionality requirements for increased demands for real-world applications, such as mechatronic systems designs, can only be realised by cost-effective and high quality design, fabrication and test activities. As such, test engineering needs to be seen as a key input to the provision of integrated circuits for systems designers.

Acknowledgements

The authors would like to thank the University of Limerick Foundation for the funding to support the work with the TRIP initiative managed by the Centre for Teaching and Learning at the University of Limerick.

References

1 R. Venkateswaran and P. Mazumder, 'A survey of DA techniques for PLD and FPGA based systems', *Integration, the VLSI Journal*, **17** (1994), 191–240.
2 N. Tredennick and B. Shimamoto, 'Special report on reconfigurable devices', *IEEE Spectrum*, (December 2003), 37–40.
3 MACH4, ispLSI 8000/V (SuperBIG), Lattice Semiconductor Corporation, http://www.latticesemi.com/
4 S. Hurst, *VLSI Testing Digital and Mixed Analogue/Digital Techniques* (IEE, Stevenage, 1998).
5 R. Rajsuman, *System-on-a-Chip Design and Test* (Artech House, London, 2000).
6 M. Burns and G. Roberts, *An Introduction to Mixed-Signal IC Test and Measurement* (Oxford University Press, Oxford, 2001).
7 I. A. Grout, 'Overview and Development of a Test Engineering Teaching Module', presented at EAEEIE 13th Ann. Intl Conf. on Engineering Education, York, UK, 8–10 April 2002.
8 Ian Grout and Joseph Walsh, 'Test Engineering Laboratories with the Lattice MACH-4 CPLD', in *Proc. Educational ECAD User Group Workshop, Buckinghamshire Chilterns University College, UK, 3–4 September 2002,* http://www.eeug.org.uk.
9 Gordon W. Roberts and Albert K. Lu, *Analog Signal Generation for Built-in-Self-Test of Mixed-Signal Integrated Circuits* (Kluwer, Amsterdam, 1995).
10 D. Bradley, D. Seward, D. Dawson and S. Burge, *Mechatronics and the Design of Intelligent Machines and Systems* (Stanley Thornes, Cheltenham, 2000).
11 H. Soemers and J. van Eijk, 'Mechatronics education at Philips', in *Proc. 8th Mechatronics Forum Intl Conf., Mechatronics 2002,* 24–26 June 2002, pp. 137–141.
12 D. Bradley, 'Mechatronics, an established discipline or a concept in need of direction?', in *Proc. 7th Mechatronics Forum Intl Conf., 6–8 September 2000.*
13 IEEE Standard 1076–1993, VHSIC Hardware Description Language (VHDL), http://standards.ieee.org/
14 IEEE Standard 1364–2001: Verilog Hardware Description Language, http://standards.ieee.org/
15 JTAG Boundary Scan, IEEE standard 1149.1, http://standards.ieee.org/
16 Joseph Walsh and Ian Grout, 'An investigation into the requirements of a PC-based learning environment for the education of microelectronic test engineering', in *Proc. 4th Annual Irish Educational Technology Users Conference (EdTech 2003), Waterford, Ireland, 22–23 May 2003.*
17 J. Coleman et al., 'Effectiveness of computer-aided learning as a direct replacement for lecturing in degree-level electronics', *IEEE Trans. Educ.*, **41**(3) (1998), 177–184.
18 C. Enloe et al., 'Teleoperation in the undergraduate physics laboratory – teaching an old dog new tricks', *IEEE Trans. Educ*., **42**(3) (1999), 174–179.
19 I. Grout and J. Walsh, 'Overview and development of a remote electronic circuit test engineering experimentation laboratory', *Proc. 2003 Interactive Computer Aided Learning (ICL2003) Workshop, Carinthia Tech Institute, Villach, Austria, 24–26 September 2003.*