AUV Localization Process using Computing Cells on FPGA

M. H. Al-Doori[1], R. Badlishah Ahmad[1], Abid Yahya[1] and Mohd. Rizal Arshad[2]

**[1] School of Computer and Communication Engineering, Universiti Malaysia Perlis, Kuala Perlis,
02000, Malaysia**

**[2] School of Electric and Electronic Engineering, Universiti Sains Malaysia, Penang,
14300, Malaysia**

Abstract

Concurrent processors incorporated on a single chip for performing computations can be troublesome because as the size of problems increase, the execution time also increases. In order to make sure that the execution time is reduced, a traditional approach is used that implies that the problem will be computed by an array of processors that are interconnected over a high-speed interconnection network. A Duplicated process is presented in this paper to handle the complexity of computing arithmetic of AUV localization operation. A single clock cycle is sufficient for performing all the operations in this process. This type of process is referred to as Duplicated because each cell incorporates a variety of duplicated multipliers and adders. The minimum number of clock cycles are needed to complete each iteration. However, each cell takes up more area on the chip. The calculation rate of the duplicated process is high despite the fact that its throughput is significantly limited by the small number of cells that can be accommodated by the FPGA.

Keywords: *Embedded system design, FPGA system design, concurrent processing, underwater detection.*

1. Introduction

Virginia Tech introduces an 'X' terascale cluster. This cluster is a clear example of this kind of parallel machine [1]. An InfiniBand network is developed by making use of 1100 Power Mac G5s. The supercomputer used for this network has successfully achieved a computation rate of 10 million operations each second, takes up a space of 280 square meters and costs $5.2 million. While these supercomputers can help to integrate massive parallelism, they are not appropriate according to some researchers who debate that using them is not a feasible approach. This is because these supercomputers are quite large and costly and work on a time-shared basis. It is because of this reason that a solution such as PCI card is used to increase hardware acceleration. These PCI cards can be incorporated into the researcher's PC and can be effectively used for computing any complex and mobile parallel problems. This ensures that the computational needs of a researcher are met effectively.

This paper introduces the design and implementation of computing process that is used on an FPGA chip. The degree to which parallelism is supported within a logic element varies from process to process and it varies also with respect to the basic methods that are used for a variety of standard arithmetic operations. As compared to the traditional approach used by parallel computers that incorporate general-purpose processors as well as software to solve any problem, such scenarios require that the hardware should be configured for problems such that the performance and size of the design are significantly maximized. In addition, support should also be provided for the hardware to be reconfigurable as this helps to execute a variety of problems on the same device. Because of the availability of FPGAs that promise high density and high performance, such parallel computing devices can be used. A small-scale processor can be effectively transformed into a cell in the mobile parallel problem by integrating programmable logic in an FPGA. Similarly, it can also transform into a parallel computer given that the cells are replicated by embedding them throughout the chip. In addition, it also helps one to determine the number of logic elements that are can easily and effectively be embedded in a single chip.

2. Underwater Localization

Leonard has provided a good survey of the various underwater localization methods [2]. Accurate motion information has been secured by acoustic sensors such as Doppler velocity. Position can be determined through integration but may lead to unrestrained increase in error. When the sensors are on the surface, Long Base Line (LBL) systems are used that integrate transponder beacons. The position of these transponder beacons is known through GPS or survey. A vehicle is used that estimates the position of the transponder on the basis of the round trip delay. The velocity of sound underwater is dependent on a number of factors such as salinity, pressure, temperature etc. and this result in uncertainty in this class of methods.

Localization making use of vision is being used by a number of researchers for navigation sensing of robots. Some useful examples include in Amidi [3] which was a detailed investigation that integrated feature tracking for an independent helicopter for image odometry. Corke is another example that employed vision for achieving stabilization in helicopter hover [4]. Dunbabin [5] that have employed underwater vision for navigation as well as station keeping [6, 7]. Other uses include station-keeping and positioning is presented by Dalgleish of cables, fish and pipes[8]. Long-term consistency in terms of motion estimate is of critical significance in such systems. Current researches are focused on motion reduction that is achieved by estimating drift in sequences of long images through techniques such as SLAM. Most of the algorithms are implemented to record data sets or made use of an ROV offering great computing power and is not just an energy constrained AUV.

Each method used has its pros and cons. Visual odometry, discussed above, naturally incorporates perceive motion and leads to long term drift. Furthermore, it requires the use of positional fixes every now and then to keep the errors restrained. Acoustic localization systems involve the development of infrastructure that is not just powered but offers accurate position estimates as well. There are hybrid systems that exist that successfully incorporate the advantages of all methods but they are not the focus of this project [9].

3. Design Flow and Development Tools

A process is described in the HDL for the design work needed for this paper. The HDL transforms the description into a form that is suitable for FPGA form and defines an interface between the development platform and FPGA. The following section defines a variety of development steps that are performed by making use of tools that are used throughout the design process. The steps are illustrated in Fig. 1, where the major steps describe the major development process. The major development process can be divided into two steps that are the supporting design and the FPGA design. Here the design flow is needed to understand the core concept in an FPGA as it gives an overview of the FPGA particularly used in this research. Before synthesizing, the design is verified for correctness. This is because synthesis may take more time than expected. Simulation and verification of waveforms over CAD tools is done for the verification of the design.

All the steps involved in the synthesis step is done using the software from Altera® as it is the manufacturer of the FPGA chip used in this design. Once the required components are defined, they can be easily be placed to the defined location of the chips and the interconnections are then maintained between them. Because of the implementation of the interconnection procedure in the device, using the place and route steps may take more time than expected. This is because the software strives to determine the routes that will satisfy the imposed timing constraints. A binary file is produced soon after the design is placed and routed. This binary file contains all the information that is required for the programming of FPGA with the design. QUARTUS® II 10.1 is used for all the research, placement, routing and programming for file-generation.

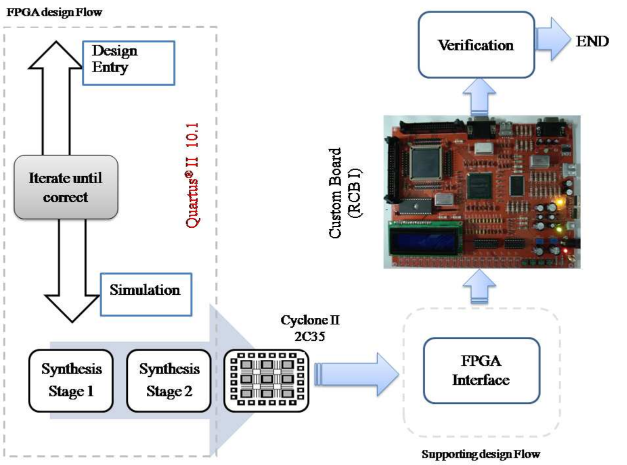


Fig. 1 Flow diagram of the major design development steps

4. Duplicated Process

The Duplicated process gets its name because every single computing cell integrates a variety of duplicate multipliers and adders such that one is used for each multiplication and addition being performed. Combination logic is used for the execution of computation as this allows fresh values to be generated every single clock cycle. However, since handshaking is used to write the computed results to memory, two clock cycles are needed per iteration. In short, the cell controller uses massive FSM that allows it to switch between the two states easily. The computation is performed during one state and during the other state, the value is stored in the memory. The Duplicated process makes use of a large area of chip so its usefulness is for the sake of comparison with other process.

Equations that have to be computed can be written readily into the description of the hardware as VHDL provides support for arithmetic operations. However, a feasible approach is to transform your equation in a form that requires the minimum number of multiply operations. A strong motivation behind it is that hardware for multiplication consumes more area on the chip as compared to the one used for addition. The localization strategy for the Duplicated hardware is illustrated in Fig. 2. All the parameters are specified and are fixed point scaled integer. This allows the results of the multiplication operations to be scaled effectively and efficiently. At first, a custom VHDL multiplication operation is performed and this renders a 2W-bit result. The illustration for the adders and multipliers units can help one to understand why so much chip area is consumed by them. An enable signal is used that allow the latching of new results into registers. The FSM in the cell controller is responsible for the generation of the enable signal.

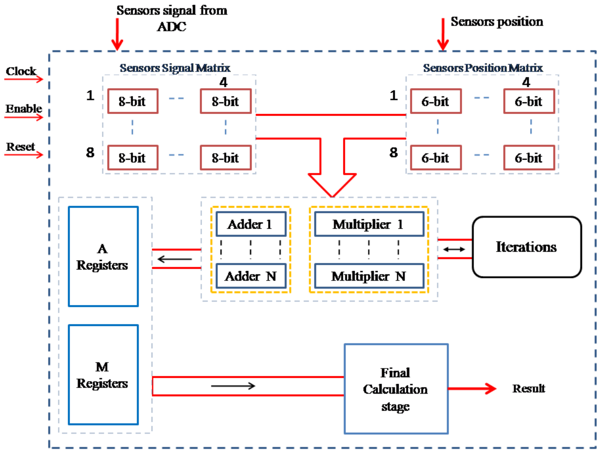


Fig. 2 Duplicated process

5. Top-Level Design

The tasks that the FPGA is responsible for performing and the hardware that will be needed is described below so as to facilitate the development of design. The steps that the FPGA perform during a complete calculation cycle are illustrated in Fig. 3. Configuration data is downloaded from the logic module of the flash memory soon after the power-up. The control module also initializes the input module at the same time and sensors then receives the input signals that are sent at the same time. Once these signals are received, they are sent over the high-performance bus at the logic module. A write signal is then forwarded to the logic module. Before computing is initiated by FPGA, all the parameters reside at the start of the logic module in a defined order that is iterations, result set and then the rest of the parameters. Iterations here is used to refer to the number of cycles it will take to complete the localization accurately. If N computing cells are used to configure an FPGA then the resulting cell will actually be a number somewhere in between 1 and N. This cell number will define where data will be stored in memory.

As far as the localization problem is concerned, the cell data will be stored at equal matrix distances each time. Furthermore, the sensor position that determines the directionality on the array of the sound sensor is used for the localization problem. Similarly, the distance applied to cell N will be the same as the distance applied to cell 1. It is important that the input parameters are stored as fixed point scalar integers as they support the number scheme that has been incorporated into the design. Once the controller sends these parameters, the FPGA receives a start signal that will determine whether or not the calculation is ready to proceed.

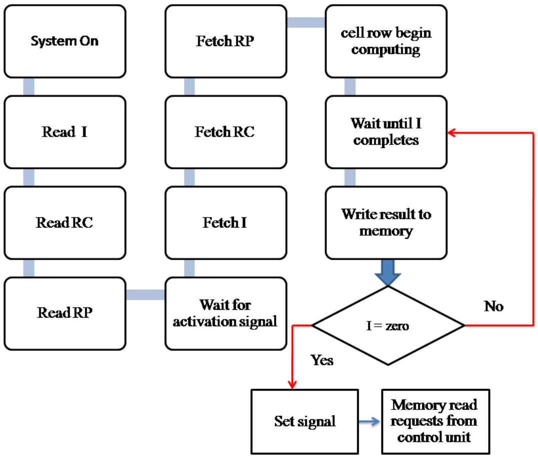


Fig. 3 Tasks performed by the FPGA design

The memory gains control of the input parameters and reads them once the FPGA receives the signals. After making sure that are parameters are in place, the computing cells arranged in a row are instructed to begin calculation and when iteration is completed, the value is stored in the resulting cell. This is because the memory starts writing the input parameters at the start of the memory space. The iteration continues to store the values in the memory until all of them are complete. Here the FPGA will release control of the memory as this allows the results computed by the control module over the bus. The new calculation cycle is initiated by the FPGA and all the initial values are then reset. If the second calculation cycle has already started, the results will be overwritten over the results from the first cycle of calculation.

Fig. 4 shows the structure of the top-level FPGA design that implements the procedure in Fig. 3. The cell row controller used is originally a Finite State Machine (FSM). FSM performs a number of tasks such as the generation of the start signal, reading the input parameters for the memory, initiating the row computation and transmitting the results back to the memory. In addition, it keeps track of whether or not the desired number of iterations have been completed. Fig. 5 is illustrating the FSM of the cell row controller. All the cells in a row are computed by a single cell FSM controller. This FSM controller is different for each process. The memory controller is not only responsible for the generation of memory chip signals but also facilitates the timing of reads and writes. The bus and row controller are both multiplexed over the memory through the data and control lines and the data from both the memory and row controller are multiplexed onto the bus. Logic is also incorporated into the design that decodes the addresses from the bus. An address space in the logic module is responsible for writing FPGA's start signal. This happens because a write signal is generated by the control module to the single address in the logic module. Similarly, a read signal is also generated for the same memory address so as to the read the row controller's signal. This instructs the control module that it can start reading the calculated results that are stored in the memory.

A separate clock is employed by the FPGA for the computing cells from the bus clock. A reason justifying its use is that the clock for the computing cells is not constrained by the bus frequency. Data is seamlessly transmitted between two asynchronous clock domains. To ensure that data is delivered properly, a handshaking sequence is performed. This handshaking is done to guarantee that all the possible combinations of cell and bus clock frequencies can be implemented.

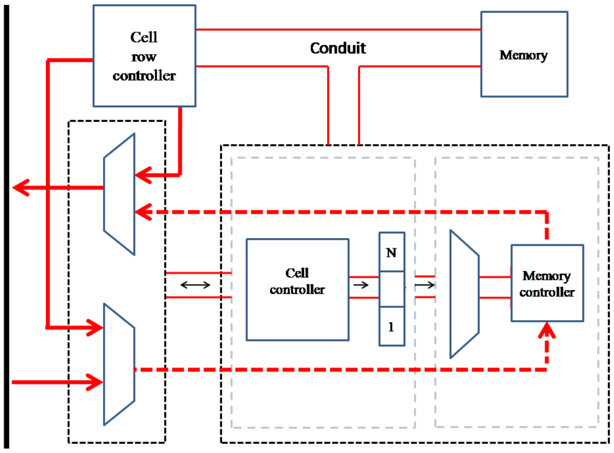


Fig. 4 Structure of the top-level FPGA design

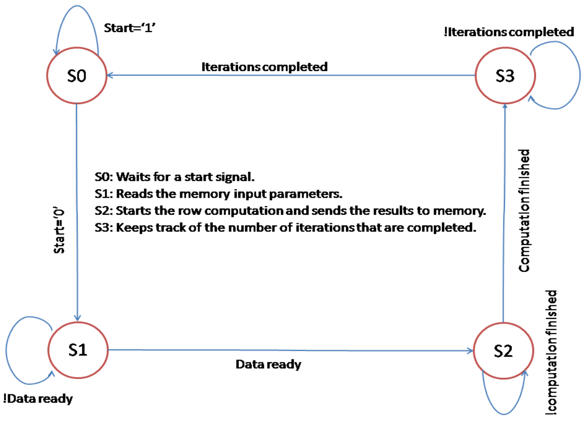

Start='1'
!Iterations completed
S0
Iterations completed
S3
Start='0'
Computation finished
S0: Waits for a start signal.
S1: Reads the memory input parameters.
S2: Starts the row computation and sends the results to memory.
S3: Keeps track of the number of iterations that are completed.
S1
Data ready
S2
!computation finished
!Data ready

Fig. 5 FSM for cell row controller

6. Localization Process

Ensuring that navigation and localization are performed reliably within the underwater environments is a challenging feat to accomplish. It is, therefore, imperative to determine the distance that the AUV has moved to get repeatable and correct measurements have been taken for implementation in other applications. Several different approaches have been proposed to get an idea about the vehicle motion as they are either acoustic based or vision based. In this study, two different processes using REP have been proposed for underwater localization. The location of the static sound sensors that are embedded on the AUV are self calibrated by the embedded localization and are responsible for covering all the views and directions. There are a total of 32 sensors and their uniform distribution over the AUV is described as follows and shown in Fig. 6:

- 6 on the right side.
- 6 on the left side.
- 6 in front.
- 6 at the back.
- 4 on top.
- 4 at the bottom.

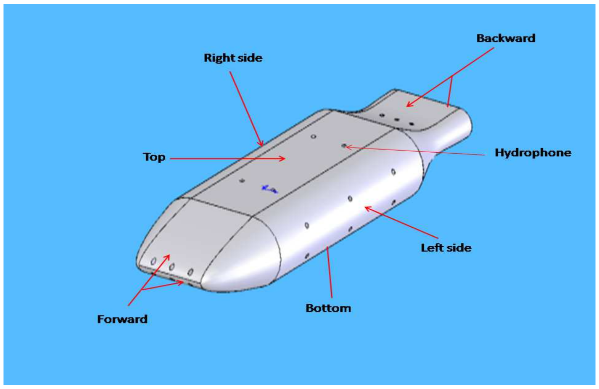


Fig. 6 Sensors arrangements on AUV [10]

The two dimensional 4x8 matrix is used to determine the style in which all these sensors are arranged. Also an 8 bit incoming represents a specific sensor's location from a fast ADC chip that is ADC08B3000. In addition, one other two-dimensional matrix 4x8 is used that specifies the location of each of the array sensors by 6-bit such that 3 bits are used for side direction and the remaining three bits are used for representing a position on a certain side. All these inputs are coming from the identification circuit of the sensor's voltage. For instance, (001000) is used to refer to the first sensor on the right side. Information regarding location is provided to the underwater vehicle through these matrices.

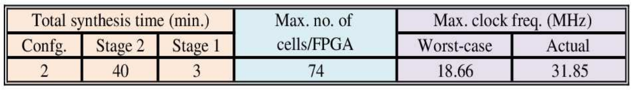
The signals generated by the array of sensors enter the 8-bit ADC at a fast speed. The ADC is responsible for converting the signals from analog to digital at a rate of 3 GSPS. Next, these signals enter the processor employing embedded localization. This allows the processor to get a fairly accurate idea of the distance, speed and directionality of the various objects in underwater environment. The system responses in real time due to the flexibility of the suggested architecture and the high speed achieved by the FPGA chip. Also, real time responses are achieved because the frequency of the crystal oscillator is much higher as compared to the speed of sound in water. The architecture proposed can also be implemented for some other media by making appropriate modifications in the FPGA design.

Quite recently, the two proposed processes have been implemented for passive localization and tracking of the AUV. This research proposes the design and implementation and evaluates the complexity and correctness through analysis and synthesis that leads to the verification of results. The acquisition of data and other types of computational processes are implemented to make use of an underwater distributed array of sensors that are embedded on the AUV. This implies that it is imperative to overcome the localization errors and the uncertainties experienced in tracking. There are several challenges that have to be addressed when it comes to tracking and localization. For example, taking range measurements concurrently is a critical operation. It is, therefore, important that the suggested architecture provides solution for the excessive motion of the AUV between the varieties of range measurements.

7. Results

The performance of the duplicated process is illustrated in table 1. The illustrations show the total synthesis time which is equal to the time taken by the QUARTUS® II 10.1 to compile design, do through the synthesis stage, and produce a program file plus the time required to transmit the logic module with the configuration data. N_{max} is equal to the total number of cells that are embedded in the chip for process. The maximum clock frequency for the worst-case at which the cells can run is also taken into consideration. The value is delivered by QUARTUS® II 10.1 after thorough synthesis as it is an average of the maximum clock frequency that is maintained while addressing all the device tolerances and taking all the operating conditions into account. Also, table shows the actual maximum clock frequency, f_{max}, achieved in the laboratory. This was determined by stepping up the clock frequency by 5MHz until the FPGA produced erroneous results.

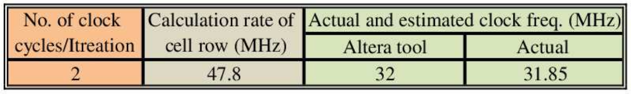
Table 1: Synthesis time, Max. no. of cell and Max. clock frequency



Total synthesis time (min.)			Max. no. of cells/FPGA	Max. clock freq. (MHz)	
Confg.	Stage 2	Stage 1		Worst-case	Actual
2	40	3	74	18.66	31.85

The operating frequency for the worst cases is obtained by employing certain latencies used through the different stages of design and observing the system's response and final throughput on the RCB I [11]. Actual operating frequencies are provide by the RCB I without using any latency injection. In order to get values of frequency that work for both cases, each operation generates a probe signal and these signals are then designated to different pins of the FPGA chip. Table 2 shows the number of clock cycles, N_{cycles}, that are needed for each iteration as specified by the number of states implemented by the FSM in the cell's controller. It also depends on the type of operation that is performed in a particular state. Also, table shows the calculation rate employed by process.

Table 2: No. of clock cycles, calculation rate, actual and estimated clock



No. of clock cycles/Itreation	Calculation rate of cell row (MHz)	Actual and estimated clock freq. (MHz)	
		Altera tool	Actual
2	47.8	32	31.85

Furthermore, the synthesis tool by Altera$_{®}$ provides the estimated values for the maximum clock frequencies achieved after the mapping process which are fairly accurate. However, they are overestimated for this particular process design. This is because of the increased routing delay which is inherent in this process because of the maximum device utilization; the routing delay is reduced by a variety of different synthesis techniques as it strives to achieve the Altera's estimated frequency.

5. Conclusion

While any number of computing cells can be accommodated and described by the VHDL design, the code is intentionally kept specific to problems so that the rate of computation for any particular problem is improved. As a result, considerable modifications in the code are also demanded by other problems. The code that offers a description of the interface between the development board and the FPGA is problem independent. Although, it does not require any modifications, extension of the design require certain changes in the problem-specific code. This paper presented design of process for mobile computing that uses FPGA as a base for the effective implementation of the realistic tracking and localization of the AUV.

References

[1] Tech, V., Virginia Tech Terascale Computing Facility, available at www.tcf.vt.edu. Access Date at Aug. 2009, from www.tcf.vt.edu, 2004.

[2] Leonard, J. J., Bennet, A. A., Smith, C. M., & Feder, H. J. S." *Autonomous underwater vehicle navigation (Technical Memo)*", MIT Marine Robotics laboratory, 1998.

[3] Amidi, O., "An autonomous vision-guided helicopter," Ph.D. dissertation, Dept of Electrical and Computer Engineering. Pittsburgh, PA: Carnegie Mellon University, 15213, 1996.

[4] Corke, P., An Inertial and Visual Sensing System for a Small Autonomous Helicopter. J. Robot. Syst., 21(2), 43-51, 2004.

[5] Dunbabin, M., Usher, K., & Corke, P., Visual motion estimation for an autonomous underwater reef monitoring robot (Vol. volume 25): Springer Verlag, 2006.

[6] Rives, P., & Borrelly, J. J., Visual servoing techniques applied to an underwater vehicle. Paper presented at the Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference.

[7] van der Zwaan, S., Bernardino, A., & Santos-Victor, J., Visual station keeping for floating robots in unstructured environments. Robotics and Autonomous Systems, 39(3-4), 145-155, 2002.

[8] Dalgleish, F., Tetlow, S., & Allwood, R., Vision-based navigation of unmanned underwater vehicles: a survey part 1: Vision-based cable-, pipeline-, and fish tracking. Journal of Marine Design and Operations, B7, 51-56, 2004.

[9] Wadoo, S., & Kachroo, P. "Autonomous Underwater Vehicles: Modeling, Control Design and Simulation", Califonia: CRC p.112-134, 2010.

[10] Salih, Muataz H., R. B Ahmad, Abid Yahya & Mohd. Rizal Arshad, "Embedded Concurrent Computing Architecture using FPGA", Proc. of the 2012 7th International Conference on System of Systems Engineering, Genoa, Italy - 16-19 July 2012.

[11] Salih, Muataz H., R. B Ahmad, Abid Yahya & Mohd. Rizal Arshad, " FPGA Design and Implementation of Multi-Filtering Techniques using Flag-Bit and Flicker Clock", IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 2, July 2012.

Muataz H. Salih received the B.Sc. and M.Sc. degrees from the Department of Computer Engineering from University of Technology, Baghdad, Iraq, in 1998 and 2002, respectively. From September 1998 to March 2003, he was a research engineer in Military Industrialization Corporation of Iraq. From October 2003 to June 2008, he was a lecturer and manager of engineering faculty's LABS in the faculty of Engineering of Al-Kalamoon private university, Derattiah, Syria. Currently, he is a Ph.D student at Universiti Malaysia Perlis (UniMap), Kuala Perlis, Malaysia. His research interests on designing digital systems using FPGA technology, embedded systems, computer system architecture, microprocessor architecture, computer interfacing, active jamming system for laser missiles, and real time systems.

R.B. Ahmad obtained B. Eng. in Electrical & Electronic Engineering from Glasgow University in 1994. He obtained his M.Sc. and PhD in 1995 and 2000 respectively from University of Strathclyde, UK. His research interests are on computer and telecommunication network modeling using discrete event simulators, optical networking & coding and embedded system based on GNU/Linux for vision. He has five (5) years teaching experience in University Sains Malaysia. Since 2004 until now he is working with University Malaysia Perlis (UniMAP). Currently as the Dean at the School of Computer and Communication Engineering and Head of Embedded Computing Research Cluster.

Abid Yahya earned his B.Sc. degree from University of Engineering and Technology, Peshawar, Pakistan in Electrical and Electronic Engineering majoring in telecommunication. Dr. Abid Yahya began his career on a path that is rare among other Researcher executives and earned his M.Sc. and Ph.D. degree in Wireless & Mobile systems, in 2007 and 2010 respectively, from the university Sains Malaysia, Malaysia. Currently he is working at School of Computer and Communication Engineering, university Malaysia Perlis (UniMAP). His professional career outside of academia includes writing for the International Magazines, News Papers as well as a considerable career in freelance journalism. He has applied this combination of practical and academic experience to a variety of consultancies for major corporations.

Mohd Rizal Arshad graduated from the University of Liverpool, in 1994 with a B.Eng. In Medical Electronics and Instrumentation. He then pursues his MSc. in Electronic Control Engineering at the University of Salford, graduating in Dec. 1995. Following from this, in early 1999, he continues with a PhD degree in Electrical Engineering, with specialization in robotic vision system,. Since then, he has been working at the Universiti Sains Malaysia (USM), Malaysia as a full-time academics, i.e. lecturer and researcher. He has supervised a number of postgraduates students at the MSc. and PhD. levels. He has also published actively in local and international publications. Dr. Mohd Rizal Arshad is currently an Associate Professor and the deputy dean of the School of Electrical and Electronic Engineering, USM. And with his team of researcher, is also the pioneer of underwater system technology research efforts in Malaysia.