An FPGA-based controller for collaborative robotics

B. P. Jeppesen, N. Roy
Intel FPGA
High Wycombe, UK
ben.jeppesen@intel.com, niladri.roy@intel.com

L. Moro, F. Baronti
Dipartimento di Ingegneria dell'Informazione
Università di Pisa
Pisa, Italy
l.moro2@studenti.unipi.it, federico.baronti@unipi.it

***Abstract*—The use of robots is becoming more common in society. Industrial robots are being developed to work with people, and lower-force collaborative robots are being developed to help people in their everyday lives. These may need fast and sophisticated motion control and behavioral algorithms, but are expected to be more compact and lower cost. This paper proposes a processor plus FPGA solution for the control systems for such robots, where the FPGA performs all real-time tasks, freeing the processor to run lower-frequency high level control and interface to other devices such as camera systems. A demonstrator robot is designed, combining multi-axis motion control with 3D robot vision.**

Keywords—Robotics; FPGA; motor; motion; control; vision; collaborative

I. Introduction

Collaborative robots are an emerging market with different needs to traditional industrial robots. For example, traditional industrial robots used for factory assembly are physically strong and rely on a working environment closed to humans for safety. A networked modular system architecture with centralized motion controller and individual motor drives is often used to control these robots [1]. On the other hand, collaborative robots may have force capability comparable to humans and be required to work around humans without injuring them [2]. These differences suggest different priorities for the control system architecture.

A single processor with an FPGA may be suitable for the main control system, reducing system cost while enabling higher frequency control loops and better synchronized control. This paper discusses the potential of this architecture for a six-axis robot controller, and describes a practical low-cost system to develop functionality using 3D vision for interaction (Fig. 1). The rest of the paper is organized as follows.

Section II surveys current technology in industrial robotics, the needs of collaborative robots and the relevance of FPGAs. Section III discusses the applications of FPGAs to robotics identified in Section II, and argues that these do not meet the likely requirements for collaborative robotics. An architecture is proposed that combines a hard processor running Linux with an FPGA to form a single controller for multi-axis robots, which meets the needs identified in Section II.

Section IV describes a low-cost hardware and software platform combining available development kits to enable development of low-cost collaborative robot demonstrator. This combines multi-axis motion control with 3D vision. Section V presents results from the low-cost demonstrator, as well as predicted results from a six-axis controller that would be suitable for an industrial collaborative robot. Section VI draws conclusions from the work so far.

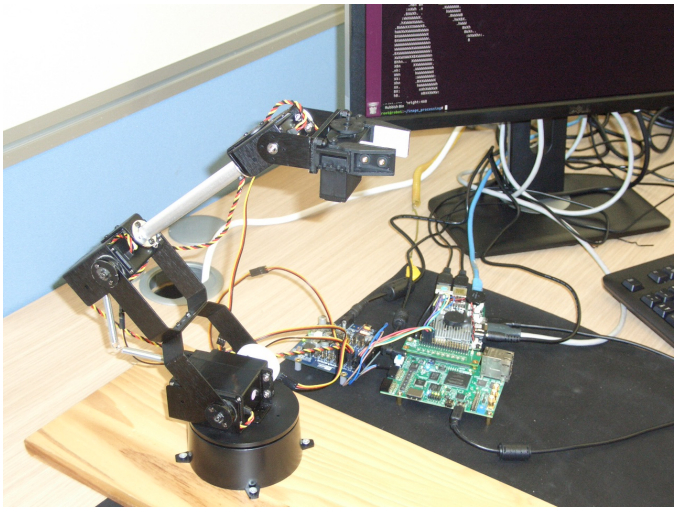

Fig. 1. Robot arm with controller boards and 3D depth info on screen

II. Industrial Robotics

A. Applications of industrial robotics

An industrial robot, as defined by ISO 8373, is 'an automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes, which may be either fixed in place or mobile, for use in industrial automation applications'. Robots can perform a variety of tasks that are too hazardous for humans to perform, too monotonous, or require speed and precision that is difficult or impossible for humans to achieve. Examples include product assembly in factories, remote working in nuclear environments, bomb disposal and medical surgery.

As robot technology has matured, manufacturers and end-users have begun to explore beyond the initial strategic advantages. Current trends in industrial robots include [3]:

- Demand for more cost-efficient robots, with high reliability and productivity
- Robots for high-performance applications such as water-jet and laser cutting, material-handling, arc-

welding, gluing, dispensing and deburring, where control strategies are different from those in assembly

- Collaborative robotics coordination, where two or more robots work on one object perhaps held and articulated by a third robot; e.g. in arc-welding
- Machine-vision guided robot control, where the desired robot trajectory is commanded by a machine-vision system, e.g. in fetching and sorting

Robotic capabilities are now being combined with additional safety features to create 'collaborative robots' that can work alongside humans to help them in their production tasks [2]. They make automation accessible to small and medium manufacturers via their low cost.

B. *Safety*

Manufacturers of collaborative robots are introducing new safety features and certifying their products to safety standards, for example:

Bionics Robotics GmbH's "BioRob Arm". BioRob arm is accredited for safe human-robot collaboration by the German Trade Association in accordance with the European Union machinery directive 2006 / 42 / EC [4].

Yaskawa has certified its collaborative robot, the Motoman HC10 (which can handle loads up to 10kg and has a reach of 1.2m), to ISO/TS 15066:2016, which specifies safety requirements for collaborative industrial robot systems and the work environment. The robot uses a force/torque sensor in each axis to help avoid dangerous collisions with operators [5].

MRK-Systeme GmbH offer a 'conversion kit' to convert a Kuka KR 5 ARC HW robot into a force limited robot. They call this Kleinroboter (small robot) KR 5 SI (for Safe Interaction). This can transform an ordinary industrial robot into a collaborative robot approved by DIN EN ISO 10218 (for general robotic devices) and eventually ISO TS 15066 regulations [4].

In additional to physical safety features, a control system must be sufficiently safe in the presence of faults. The general industrial functional safety standard, IEC 61508, exists to help the design of programmable electronic systems used in industrial machines which could cause a hazard to human life.

C. *FPGA SoCs for industrial control*

System-on-Chip (SoC) FPGAs, which combine a processor and FPGA fabric on a single chip, offer these advantages for control system designs [6]:

- High-performance control - SoC FPGAs can off-load the processor by implementing digital signal processing (DSP) algorithms in the FPGA. Implementing in hard logic gives repeatable execution times and minimizes latency, maximizing the controllable bandwidth of real-time control systems [8]. Computationally-intensive control algorithms like sensorless motor control [12], direct torque control [13] and model-predictive control [9][10][11] can be enabled by FPGAs. The hardware description language code can be generated automatically from languages including C [9] and Simulink [8]. Some FPGAs now include hardened floating point DSP blocks [7], reducing algorithm development time and hardware execution time compared to converting floating point calculations to fixed-point.
- Connectivity—SoC FPGAs can implement multiple Industrial Ethernet protocols, including emerging standards, such as IEEE 802.1 TSN, simultaneously on a single device by instantiating ready-made intellectual property (IP) cores. The relevant protocol stacks execute in the built-in SoC FPGA Hard Processor System (HPS). The high-performance FPGA fabric can easily meet the stringent IEEE 802.1 TSN timing requirements [6]. The HPS can also run an OPC server, enabling enterprise communications over OPC-UA.
- Secure communications—Open SSL encryption, implemented in the FPGA fabric, provides acceleration over processor-based implementations. This encryption enables secure enterprise communication channels [6].
- Future proofing—Designers can reprogram the FPGA fabric, avoiding major redesign of entire systems.
- Functional Safety: Manufacturers such as Intel FPGA provide programming tools and IP certified to functional safety standards including IEC 61508 and ISO 26262. The flexibility of the FPGA allows the designer to build in additional redundant logic or safe communication protocols.

D. *FPGAs SoCs for accelerating DSP in robotics*

In [14], kinematic calculations are implemented in an FPGA for faster execution. The 'Motion controller IP' also includes 5-axis speed and position control loops, parallel quadrature encoder interfaces and PWM generation blocks. The motors are 24 V geared DC motors. A similar FPGA-based motion controller for DC motors is used in [15], with the addition of image processing IP from stereo CMOS image sensors. A further example is [16], where nonlinear adaptive and 'computed torque' algorithms are partitioned between a DSP and FPGA. [16] demonstrates the FPGA's capability for high frequency control loop updates, achieving 120 kHz current control and 20 kHz velocity control.

III. Concept for an FPGA-based collaborative robot control Architecture

Section II showed that FPGAs have been applied to the acceleration of kinematic calculations, motor control and parallel interfacing to multiple motors, as well image processing. However, these applications to robotics used relatively large FPGAs [14]-[16]. They targeted geared DC or brushless DC motors with Hall sensors. Faster and more accurate position control can be achieved using sinusoidal PMSM motors with position encoder feedback; these require more complex control such as field-oriented control.

The FPGA provides advantages for real-time motion control and real-time algorithm acceleration, but can be a more expensive solution than a hard processor for more common, slower or non-real-time processing. For example, it is useful to

have a processor running Linux to use open-source libraries such as OpenCV for image processing or the Robotic Operating System for robot motion planning [26].

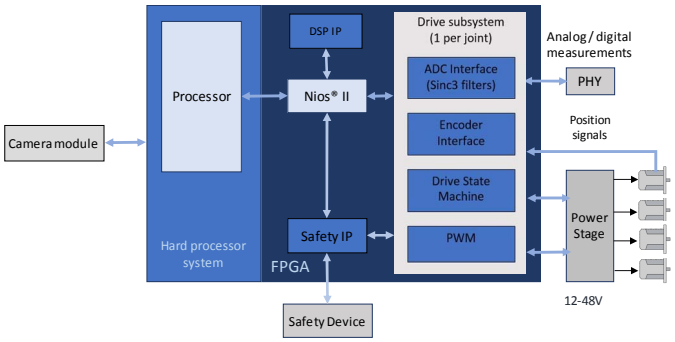

Fig. 2. Proposed controller architecture

The proposed control system (Fig. 2) comprises a hard processor system and an FPGA, which includes a Nios II soft processor that performs these functions:

- It runs real-time software tasks, freeing the hard processor to run Linux and standard software libraries.
- It is a central data master in the system, communicating with the hard processor and other IPs in the FPGA.
- It provides a place to do centralized tasks like start-up or shut-down sequences and kinematic calculations.
- It can reuse motor control DSP IP multiple times to do control calculations for each motor axis in turn.

The Nios II is a small IP which enables more efficient use of the hard processor and the FPGA. Thus, a lower cost FPGA and hard processor system can be selected compared to a single hard processor and FPGA with parallel motor control IP.

It is assumed that the robot is equipped with position and current sensors, which provide real time feedback for control. There is also a camera module to provide 3D image sensing for interaction with environment. A safety device can include separate sensors and shut down the system independently of the main processor and Nios II (Fig. 2).

The FPGA system includes floating point Custom Instruction IP, which is used like a co-processor to the Nios II. The C compiler automatically uses the Custom Instruction IP to accelerate floating point instructions in the C code. It also includes IP designed with Intel FPGA's DSP Builder Simulink library to accelerate the field-oriented control used for each motor control axis.

IV. Robotics Demonstration System

To demonstrate a practical FPGA-based robot controller, a kit has been created from off-the-shelf robot, servo driver and FPGA control hardware. The emphasis is on combined motion control and interaction based on camera input. To create a compact and low-cost demonstration, an educational robot arm has been chosen which uses self-contained servo actuators.

The robot behavior is a simple example of a collaborative robot; the robot arm follows the movement of a human hand. The hand is assumed to be the closest object to the RealSense camera (Fig. 3).

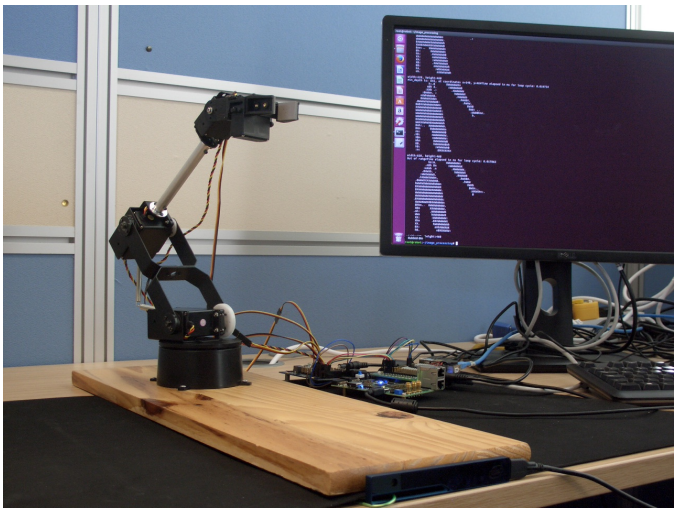

Fig. 3. Robot arm with control boards, RealSense™ camera (in front of the wood base) and screen showing 3D depth image in text from

A. Hardware

Robot and control hardware has been chosen as follows:

1) Robot arm and servos

The Lynxmotion AL5D from RobotShop [18] was selected due to its widespread use in educational robotics projects. It has 5 degrees of freedom: base rotation, shoulder joint, elbow, wrist and gripper (Fig. 3). Position commands are given to the servos in the form of 0-5 V PWM signals with 50 Hz frequency and 2.5%-12.5% duty cycle. Internally, the servos use geared DC motors with integrated potentiometer-based analog feedback control to achieve the demanded position.

An alternative educational robot which uses servos that also provide feedback signals and current-limiting capabilities is the PhantomX Reactor Robot Arm Kit from Interbotix [19]; it uses Dynamixel servos [20].

2) Servo driver board

A board is needed to provide PWM data and 5 V power to the robot servos. Two boards were considered:

- Adafruit 16-Channel PWM / Servo HAT [21], which is designed to fit above a Raspberry Pi controller with the its 40-pin connector. The interface is I2C.
- The Terasic Servo Motor Kit [22] drives up to 24 servos, the inputs being PWM signals supplied in parallel over the 40-pin connector. The board also has safety features provided by a built-in CPLD and power monitor chip.

Either board can provide appropriate PWM servo commands. However, the I2C protocol limits the resolution of the position commands to around 400 values, whereas the Terasic board receives parallel channels of PWM generated in FPGA IP, enabling much higher resolution (100,000 values). The parallel PWM architecture is also scalable to industrial systems. The Terasic board was therefore chosen.

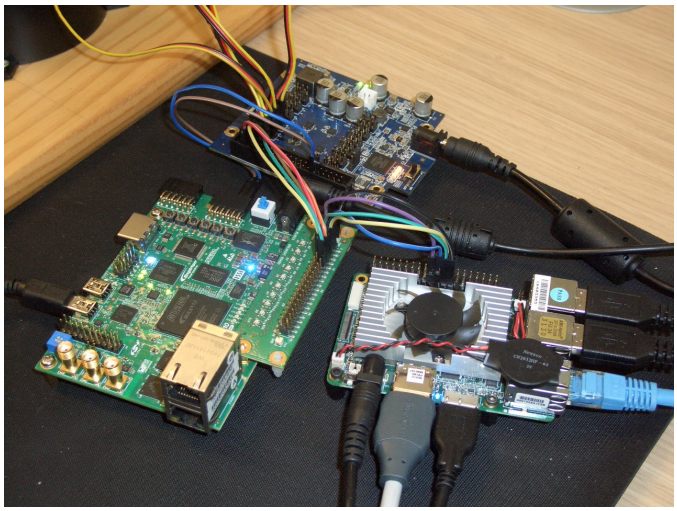

Fig. 4. MAX 10 Development Kit (bottom left) with 40-pin adaptor board, Aaeon UP board (bottom right) and Terasic Servo Motor Kit (above)

3) FPGA control board

A MAX® 10 Development Kit [24] was used, due to its flexibility, low cost and use in existing motor control reference designs [8]. Connections to the servo boards' 40-pin connectors can be made from its 'pmod' connectors or by using an expansion board to convert its HSMC connector to a 40-pin connector [23] (Fig. 4).

4) Vision processing and supervisory control board

The Intel® RealSense™ Robotic Development Kit [25] provides vision-based feedback to affect the motion of the robot arm. The kit comprises a RealSense camera with optical and depth-sensing infrared measurements [27], communicating over USB 3 with an Aaeon UP board, which contains an Intel Atom processor and standard 40-pin connector (Fig. 4). Tutorials for the board, using Ubuntu Linux and the Robotic Operating System (ROS) C Library are freely-available [26].

B. Control architecture

1) Control loops and update rates

The control loops are limited by the camera frame rate, 30Hz (frame/s), and the servo control PWM frequency, 50 Hz (Fig. 5). The camera frame rate is typical of industrial systems. The PWM frequency is much slower – typical industrial systems use around 10 kHz PWM. However, the architecture chosen would support much higher PWM frequencies.

The Intel processor software loop waits for each new frame, finds the closest point and sends it over SPI to the FPGA SPI IP. The IP accepts the data and triggers an Interrupt Service Routine (ISR) in the Nios software to buffer the data. The Nios software main loop timing is controlled by an interval timer IP which creates interrupts at 50Hz. The interrupts trigger a separate ISR, which writes position and speed commands to the PWM generating IP, and provides a count of the interrupts to the main loop. The main loop uses the count to schedule two slower (5 Hz) tasks such that they do not interrupt each other. The first calculates four new target joint positions based on the average of the last eight hand positions, while the second calculates a speed trajectory for the next ten 50 Hz speed commands. While 5 Hz is relatively slow, the software structure serves as a model for faster systems where the trajectory would be calculated at a slower rate than the PWM update frequency, and ensures smooth motion of the robot arm.

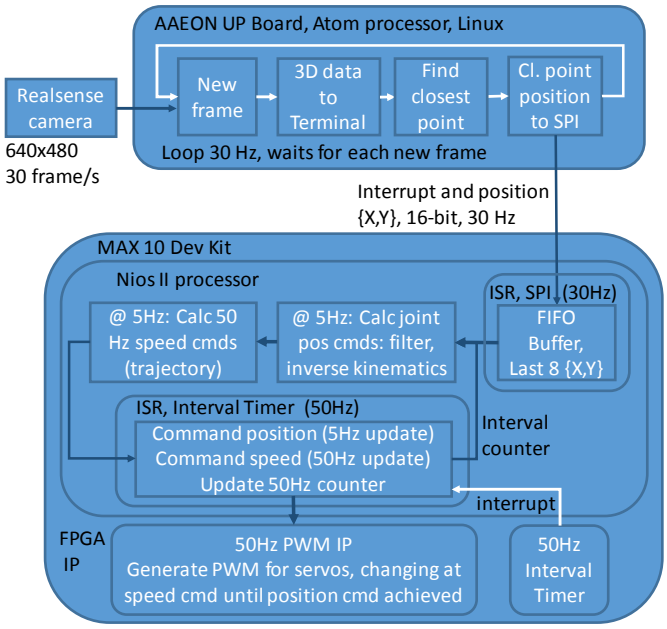

Fig. 5. Control architecture and loop updates in demonstrator system

2) Trajectory planning

Trapezoidal trajectories are calculated: 200 ms periods are split into 100 ms of constant acceleration followed by 100 ms of constant speed. The motion finishes at the next target position. Speed updates are at 50 Hz but position updates are at 5 Hz; there are 5 values of changing speed and 5 values of constant speed between each position update and the next (Fig. 6).

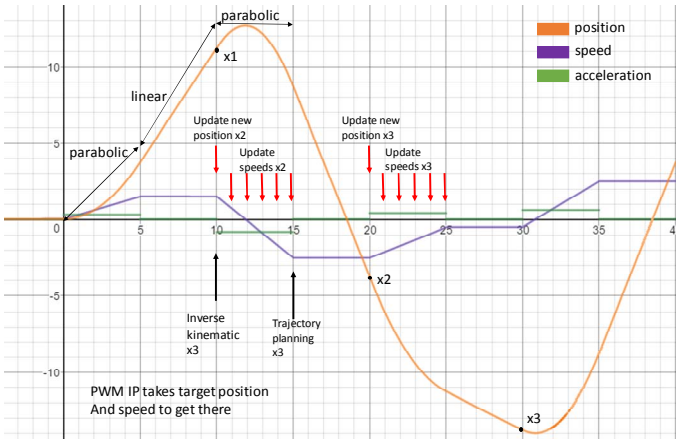

Fig. 6. Trapezoidal trajectory planning: x-axis is time in 50Hz timesteps; y-axis is position, speed and acceleration (notional scale)

V. Results and Discussion

This section presents actual results from the demonstrator system, together with simulated results from a 6-axis system representative of a state-of-the-art industrial system.

A. Execution times

1) Demonstration System

The FPGA IP runs continuously and deterministically at high clock rates (Fig. 5). Apart from the camera and PWM update frequencies, software execution times are the limiting factor in performance.

The closest point execution algorithm was measured using software timers, and took 0.025 ms. This is very small compared to the 30 Hz frame update rate (equivalent to 33 ms).

Execution times in the Nios software are shown in TABLE I. The kinematics and trajectory planning calculations both take much more time than the ISRs. Even so, their total time is less than 1 ms, so could be used to update the robot position and speed commands at up to 1 kHz, much faster than the 5-50 Hz used in the demonstrator, and suitable for an industrial system.

TABLE I. ***MAX***® 10 FPGA WITH NIOS II 100MHZ PROCESSOR, SOFTWARE EXECUTION TIMES FOR 6 FIELD-ORIENTED MOTOR CONTROL AXES

Software functions in order of execution	Time, all axes [µs]	Time / motor axis, depending on FOC [µs]			
		SW Fix	SW Float	HW Fix	HW Float
ΣΔ ADC, encoder settling time	10				
ISR service time	1				
Read ADCs and encoder		2	2	2	2
Rescale position, calc speed		1	1	1	1
Position and speed control		2	2	2	2
Field-oriented control (FOC)		4	6	1.5	3
SVM PWM		1.5	1.5	1.5	1.5
Write diagnostic data	2				
Total execution times [µs]					
Sum, overheads for all axes	**13**				
Sum, per axis times		**10.5**	**12.5**	**8**	**9.5**
Number of axes		6	6	6	6
Sum, per axis times * N axes		**63**	**75**	**48**	**57**
Totals including overheads		**76**	**88**	**61**	**70**

2) Industrial 6-axis controller

In this proposed design, the Nios II processor reads in all data, does the motor control calculations on each axis in turn, and then writes general diagnostic data such as logged signals. The field-oriented control calculations can be implemented in different ways, such as floating point or fixed point software in the Nios processor, or floating point or fixed point calculations as FPGA IP. Each choice represents a different compromise in terms of execution speed, FPGA resource usage and execution speed. The total execution time for all six motor control axes can be estimated based on results for 1 and 2 axes in the existing Drive on Chip reference design [17]. To update control at 16 kHz, the total time budget is 62.5 µs.

TABLE I. shows how the total time for six motor control axes varies depending on the implementation method of the field-oriented control. The hardware (FPGA IP) fixed-point implementation must be chosen for the time to be under 62.5 µs. This IP is therefore included in the design in TABLE III.

B. FPGA Resource Usage

The MAX 10 FPGA has limited logic resources. The main resource is logic elements, and there are also limited numbers of 'hard' DSP blocks and M9K memory blocks, which are used to accelerate calculations compared to using 'soft' logic alone.

TABLE II. FPGA RESOURCE ESTIMATION THE DEMONSTRATOR SYSTEM

Entity	**Logic Elements**	**M9K memory blocks**	**9x9 DSP blocks**
Nios II processor	3119	28	6
Floating Point CI IP	2247	3	9
DDR3 RAM interface	4809	9	0
Interconnect	4578	8	0
JTAG debug	887	1	0
Interval Timer	179	0	0
SPI interface	133	0	0
I2C interface	304	0	0
PWM IP 0	619	0	0
PWM IP 1	611	0	0
PWM IP 2	614	0	0
PWM IP 3	608	0	0
PWM IP 4	616	0	0
Other components	1173	0	0
Total	**20497**	**51**	**15**

1) Demonstration System

TABLE II. shows resources used by the IP in the demonstrator design. The total logic is less than 50% of the MAX 10's 50,000 logic elements.

TABLE III. FPGA RESOURCE ESTIMATION FOR 6-AXIS DRIVE ON MAX 10

Entity	**LE**	**M9K**	**DSP**
Nios II 32bit soft CPU	3400	28	6
Floating Point CI IP	2300	4	10
Tightly Coupled Memory	1	16	0
DDR3 RAM interface	5200	11	0
Memory-mapped interconnect	4500	0	0
JTAG debug	1000	3	0
FOC accelerator, fixed point	2400	2	24
Drive 1 subsystem	3600	0	0
Drive 2 subsystem	3600	0	0
Drive 3 subsystem	3600	0	0
Drive 4 subsystem	3600	0	0
Drive 5 subsystem	3600	0	0
Drive 6 subsystem	3600	0	0
SPI interface to ext processor	500	0	0
Safety IP	2000	6	0
Total	**42901**	**70**	**40**
Available	**49760**	**182**	**288**
Utilization	**86%**	**38%**	**14%**

2) Industrial 6-axis controller

Using the existing multi-axis motor control reference design as a reference [17], the FPGA resource usage on the MAX 10 FPGA for a 6-axis robot with field-oriented control of

PMSM motors was estimated (TABLE III.). The design fits comfortably within the 50,000 logic element device.

VI. Conclusions and Further work

This paper:

- Proposes a hard processor plus FPGA as a low-cost controller for collaborative robotics.
- Proposes the use of the FPGA for real-time calculations, so the hard processor can run Linux and use C libraries.
- Uses the example of a 6-axis motor controller doing field-oriented control at 16 kHz on a MAX 10 FPGA to demonstrate the practicality of this approach.
- Describes practical low-cost hardware that has been used to create an FPGA-based robot demonstrator with collaborative, vision-based behavior.

This work provides a platform for further development with more sophisticated robotic control or servo motor control.

Acknowledgments

Intel FPGA would like to acknowledge the support of Intel colleagues Roi Ziss and Soliman Nasser with the Intel® RealSense™ Robotics Development Kit and Fabrizio Del Maffeo at Aaeon, creater of the UP board used in the kit.

References

[1] Article from Drives and Controls on modular robot system by Nexcom, 2016: http://drivesncontrols.com/news/fullstory.php/aid/5134/Modular_robot_control_system_allows_users_to_mix-and-match.html

[2] ABB YuMi collaborative robot datasheet, 2015: https://library.e.abb.com/public/55362813a776464383279a729b715c89/ROB0317EN_YuMi.pdf

[3] N. Roy, “Industrial Robotics – Trends, Challenges and Opportunities”, EDN China, January 2016, published online in Chinese: http://archive.ednchina.com/www.ednchina.com/ART_8800523978_20_35479_TA_1bc42928.HTM (English version available from Intel FPGA or the authors)

[4] Robotiq, “Collaborative Robot Ebook”, Sixth Edition, http://robotiq.com/wp-content/uploads/2015/03/Review-of-collaborative-robot-kuka-baxter-universal-robot-abb-F.pdf

[5] Drives and Controls article on Yaskawa Motoman collaborative robot: http://drivesncontrols.com/news/fullstory.php/aid/5212/Yaskawa_unveils_its_first_collaborative_robot_outside_Japan.html

[6] Intel FPGA, White Paper 01259: “PLC Architecture in the Industry 4.0” 2015. https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/wp/wp-01259-plc-architecture-in-the-industry-4.0-world.pdf

[7] Arria 10 FPGA Handbook: https://www.altera.com/content/dam/altera-www/global/en_US/pdfs/literature/hb/arria-10/a10_handbook.pdf

[8] B P Jeppesen, A Crosland, "An FPGA-based platform for integrated power and motion control," IECON 2016, IEEE, Florence, Italy, 2016.

[9] K.V. Ling, S.P. Yue and J.M. Maciejowski, “A FPGA Implementation of Model Predictive Control”, Proceedings of the 2006 American Control Conference, Minneapolis, Minnesota, USA, June 14-16, 2006

[10] Adam Mills, Adrian Wills, Steven Weller, and Brett Ninness, "Implementation of Linear Model Predictive Control using a Field Programmable Gate Array," IET Control Theory & Applications, 2012, Volume: 6, Issue: 8, Pages: 1042 - 1054, DOI: 10.1049/iet-cta.2010.0739 University of Newcastle, Callaghan, NSW 2308, Australia. sonic.newcastle.edu.au/reports/Document915.pdf

[11] Alfonso Damiano, Gianluca Gatto, Ignazio Marongiu, Aldo Perfetto, and Alessandro Serpi, "Operating Constraints Management of a Surface-Mounted PM Synchronous Machine by Means of an FPGA-Based Model Predictive Control Algorithm," IEEE Transactions on Industrial Informatics, February 2014.

[12] L. Idkhajine, E. Monmasson, A. Maalouf, “Fully FPGA-Based Sensorless Control for Synchronous AC Drive Using an Extended Kalman Filter”. IEEE Transations on Industrial Electronics, Vol. 59, No. 10, October 2012.

[13] T. Sutikno, N. R. N. Idris, A. Jidin, M. N. Cirstea, “An Improved FPGA Implementation of Direct Torque Control for Induction Machines”, IEEE Transactions on Industrial Informatics, 2013, Vol. 9, Issue 3, pp. 1280-1290, DOI: 10.1109/TII.2012.2222420

[14] Ying-Shieh Kung, Kuan-Hsuan Tseng, Chia-Sheng Chen, Hau-Zen Sze and An-Peng Wang, "FPGA-Implementation of Inverse Kinematics and Servo Controller for Robot Manipulator," IEEE, International Conference on Robotics and Biomimetics, 2006.

[15] Yi-Ting Chen, Ching-Long Shih and Guan-Ting Chen, "An FPGA Implementation of a Robot Control System with an Integrated 3D Vision System," Smart Science, 2016. ISSN: (Print) 2308-0477 (Online) Journal homepage: http://www.tandfonline.com/loi/tsma20. Link to article, http://dx.doi.org/10.1080/23080477.2015.11665643

[16] Xiaoyin Shao; Dong Sun, Development of a New Robot Controller Architecture with FPGA-Based IC Design for Improved High-Speed Performance, IEEE Transactions on Industrial Informatics, Year: 2007, Volume: 3, Issue: 4. Pages: 312 - 321, DOI: 10.1109/TII.2007.912360. IEEE Journals & Magazines

[17] Altera (now Intel FPGA), Application Note AN-669: “Drive-On-Chip Reference Design”, 2015, www.altera.com/en_US/pdfs/literature/an/an669.pdf

[18] Lynxmotion AL5D Robot Arm from RobotShop: http://www.lynxmotion.com/c-130-al5d.aspx

[19] Interbotix PhantomX robot arm using Dynamixel servos: https://interbotix.com/p/phantomx-ax-12-reactor-robot-arm.aspx

[20] Dynamixel servos from Robotis: http://www.robotis.us/dynamixel/

[21] Adafruit 16-Channel PWM / Servo HAT: https://www.adafruit.com/product/2327

[22] Terasic Servo Motor Kit: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&No=1028

[23] Terasic HSMC to 40pin adaptor board: http://www.terasic.com.tw/cgi-bin/page/archive.pl?Language=English&CategoryNo=67&No=322

[24] MAX 10 Development Kit: https://www.altera.com/products/boards_and_kits/dev-kits/altera/max-10-fpga-development-kit.html

[25] Intel® RealSense™ Robotic Development Kit: https://software.intel.com/en-us/realsense/robotic-development-kit

[26] Tutorials for Intel® RealSense™ Robotic Development Kit: https://01.org/developerjourney/recipe/intel-realsense-robotic-development-kit

[27] RealSense camera specifications: https://software.intel.com/en-us/articles/intel-realsense-data-ranges