computers

MDPI

Article

COTS-Based Real-Time System Development: An Effective Application in Pump Motor Control

George K. Adam [1,*], Nikos Petrellis [2], Panagiotis A. Kontaxis [3] and Tilemachos Stylianos [1]

1 Department of Digital Systems, University of Thessaly, 41500 Larisa, Greece; tilemaxos@uth.gr
2 Department of Electrical and Computer Engineering, University of Peloponnese, 26334 Patra, Greece; npetrellis@uop.gr
3 Department of Electrical & Electronics Engineering, University of West Attica, 12244 Athens, Greece; pkont@uniwa.gr
* Correspondence: gadam@uth.gr; Tel.: +30-2410-684596

Received: 25 November 2020; Accepted: 4 December 2020; Published: 5 December 2020

Abstract: The progress of embedded control systems in the last several years has made possible the realization of highly-effective controllers in many domains. It is essential for such systems to provide effective performance at an affordable cost. Furthermore, real-time embedded control systems must have low energy consumption, as well as be reliable and timely. This research investigates primarily the feasibility of implementing an embedded real-time control system, based on a low-cost, commercially off-the-shelf (COTS) microcontroller platform. It explores real-time issues, such as the reliability and timely response, of such a system implementation. This work presents the development and performance evaluation of a novel real-time control architecture, based upon a BeagleBoard microcontroller, and applied into the PWM (pulse width modulation) control of a three-phase induction motor in a suction pump. The approach followed makes minimal use of general-purpose hardware (BeagleBone Black microcontroller board) and open-source software components (including Linux Operating System with PREEMPT_RT real-time support) for building a reliable real-time control system. The applicability of the proposed control system architecture is validated and evaluated in a real case study in manufacturing. The results provide sufficient evidence of the efficiency and reliability of the proposed approach into the development of a real-time control system based upon COTS components.

Keywords: real-time; motor control; commercial off-the-shelf components; BeagleBone Black; Linux; reliability; efficiency

1. Introduction

Efficient control approaches and motor controllers have always been used in engineering systems. In the last few decades, advances in microelectronics technology have made it possible to build new devices, which support complete or partial control, based on numerous control architectures and methodologies. The latest progress and achievements show that the majority of such systems for motor speed regulation and control is based upon microprocessors and microcontrollers. A considerable amount of such control systems is built upon 8-bit, 16-bit, and 32-bit processors [1–4]. A common feature is the miniaturization, which results in the decrease of the controller's cost and dimensions, as well as in the increase of its functionality.

The most efficiently used controllers proved to be the adjustable frequency alternating current (AC) controllers [5,6]. Such controllers include a pulse width modulated (PWM) driver. PWM is one of the primary digital methods for direct current motor control, in which the pulse duration of the motor's voltage is modulated by varying the duty cycle, which is the fraction of time that the output voltage

is ON compared to when it is OFF (series of ON-OFF pulses). The PWM driver typically consists of a diode convertor, since the PWM method requires constant voltage (intermediate direct current link). The PWM inverter is controlled by a microprocessor or microcomputer controller, implementing in software the pulse width modulation method [7]. The digital realization of the PWM method improves flexibility of the controller, reduces design time, and allows more precise control compared to conventional analog controllers.

The research on real-time control of time sensitive systems, particularly in manufacturing, has gained great interest in the last years. A large number of applications have been proposed in different areas of real-time control [8–11]. However, the majority of motor control systems are based on specialized and specific purpose components and less general-purpose ones [12–14]. A considerable amount of such works is based on various microcontrollers [15] and, particularly, PIC microcontrollers. For example, in Reference [16], a PIC 18F4520 microcontroller is used to realize the control of the PWM inverter. Another relative work [17], uses the PIC18F4431 microcontroller to realize the speed control of a three-phase induction motor. FPGAs (Field Programmable Gate Arrays) have also been used extensively in rapid and low-cost development of such controllers. For example, in Reference [18] realized an FPGA-based PWM controller of an induction motor drive. Other specific purpose devices and even host PCs have been used in overall monitoring and PWM-based speed control of such systems [19,20]. Generally, in industry, the development architectures and designs are built upon specialized or custom-built hardware/software components [21]. However, due to development and maintenance costs, this tendency is inclined to change towards more adaptable, low-cost general-purpose components [22–26].

COTS (commercial off-the-shelf) as a term describes software or hardware products that are ready-made and available for sale to the general public. The Federal Acquisition Regulation (FAR) (www.acquisition.gov) (accessed on 20 September 2020) has defined "COTS" as a formal term for commercial items, including services, available in the commercial marketplace that can be bought and used under government contract. COTS components are already used in real-time systems with low criticality, but they are not yet typically employed in safety-critical hard real-time control applications. On the other hand, the reliability of such systems is crucial. At present, their reliable performance particularly in safety-critical control systems is still under intense research [27–29]. A trend to use commercially available off-the-shelf components has emerged due to the immediate availability of such components, their lower cost, and energy consumption [30,31]. In addition, selecting COTS components speeds up the design and development phase. This significantly affects the overall cost of the system and the maintenance cost.

This research investigated the performance reliability aspects of using commercial-off-the-shelf hardware and software components in real-time control of a pump system of a lime milk machine. Many microcontroller manufacturers (e.g., NXP Technologies, STMicroelectronics, Texas Instruments), vendors, and organizations (e.g., BeagleBoard) have committed to providing and promoting COTS products focused in real-time control applications. The BeagleBoard microcontroller series of products have emerged as such candidates for many of the embedded computer systems of today. BeagleBone Black microcontroller's features are used as a basis of performance analysis for this research, especially with respect to the control of a pump's AC induction motor.

Upon this BeagleBone Black microcontroller board (referred from now on as BBB), the control system for the operation of a lime milk machine's pump induction motor was developed. The motor's speed control is based upon the PWM method. This is achieved by the software generation of the pulse width modulated signals, which drive the inverter and produce a near sinusoidal current in the motor's windings. This microcontroller-based control system provides a simpler and more reliable digital speed control and adds flexibility for the developer to even customize a design for a specific application. The experimental results on the actual machine under control reconfirm the applicability of the followed methodology and approach, as well as demonstrate that such a COTS-based control system (upon BeagleBone Black microcontroller board) performs relatively well compared to a previous

custom-built solution (based upon MC68705R3 Motorola single chip microcontroller). A closed-loop diagram of the proposed COTS-based approach implemented in a pump control system is illustrated in Figure 1.

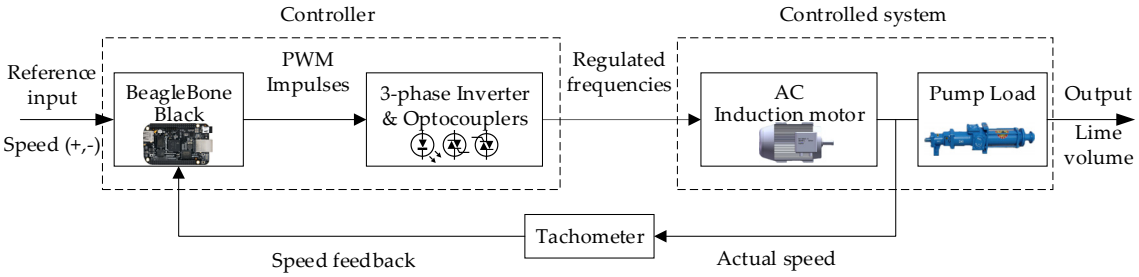

Figure 1. Overview of the commercially off-the-shelf (COTS)-based approach implementation in a pump control system.

An aspect of our research approach targeted the minimization in the use of specialized hardware/software components in real-time systems by maximizing the integration of general-purpose and open source COTS components, as well as taking in consideration their availability and reliability to develop effective systems. The goal is to bring the use of COTS hardware and software components to a practical use in an industrial context. This is because we are interested in finding answers to industrially relevant questions, such as: How to assess suitability and efficiency of a COTS component for a real-time application? How to achieve efficient design of a real-time control architecture based on COTS components? How to compose multivendors' components to make a control system suitable for real-time applications in industry?

With respect to such issues, this research work contributes on providing experimental and comparative results on real-time control of an industrial machine unit based on an embedded COTS device (BBB) and real-time open source software and OS kernel. Some of the key features and contributions of this research work are the following:

- Definition of a novel application control infrastructure based on an open system architectural framework (VIDAF [32]) that uses COTS components in building reliable and cost-efficient control systems.
- Design and implementation of a novel real-time application control architecture built upon a COTS (general-purpose) ARM-based microcontroller board (BeagleBone Black).

The outcome of this research is a flexible, low-cost embedded system prototype applied in the control of a three-phase induction motor in a suction pump. This prototype represents an added value towards the goal of using COTS components in real-time control applications. The initial findings of this research are very promising and support the concepts and results presented in Reference [32]. The research outcomes are expected to be worthy and have a positive impact on emerging areas, such as Industrial Internet of Things (IIoT) and embedded control systems based on COTS components. It is expected that the same architectural framework and methodology could be applied on other case studies in industry.

The application domain is a Machines Constructions Company (www.adam.com.gr) (accessed on 29 October 2020). The research was carried out under company's work program for research into new techniques for the design and development of control systems based on novel low-cost systems and technologies. Although this research focuses on embedded real-time control of a pump system's motor, much of its content is applicable to other manufacturing and industrial applications.

This paper is structured as follows: Section 2, Materials and Methods, describes the materials, the design methodology, and the proposed COTS-based architecture of the control system; Section 3, Results, presents the implementation of the proposed system architecture in the control of a pump's induction motor, the PWM software control module developed, and the experimental results obtained;

Section 4 provides a brief discussion of the experimentations carried out and system's overall performance; and Section 5 draws conclusions.

2. Materials and Methods

2.1. Real-Time Control Systems

A hard real-time system guarantees that critical tasks complete on time. Hard real time systems use preemptive task scheduling, so that critical tasks get immediately scheduled, whereas soft real time systems use non-preemptive priority scheduling. Real-time control systems have adopted techniques, such as high-performance multi-processor multi-threaded systems and operating systems with real-time support, to achieve the desired real-time control objectives. The standard Linux kernel successfully handles soft real-time requirements. The real-time preemption patch PREEMPT_RT utilized in this research allows nearly all of the kernel code to be preempted by higher priority kernel tasks (as threads) and reduces the maximum thread switching latency. Therefore, this patch transforms Linux into a hard real-time operating system with deterministic and predictable behavior. Nevertheless, it does not provide full assurance for hard timing deadlines required in safety critical applications in industrial automation and control. An issue that was investigated by this research, too.

Real-time control systems consist, additionally, of many different peripheral units (e.g., sensors, actuators) that need to communicate and acquire data in real-time. For this purpose, real-time systems, with strict timing constraints, need to consider several performance factors, among which is the worst-case latency or response time. Beyond such timing constraints, when selecting the COTS components, other general selection criteria are applicability, availability, low-price, and low-power. Therefore, the overall software and hardware architecture should be well-designed to enable the development of real-time applications that fulfill the real-time control specifications.

2.2. Design Methodology

Real-time embedded applications vary in size and scope: from micro-robotics to heavy-industry automation, unmanned undersea and aerial devices to aerospace industry. The wide range of applications requires a more general open framework to integrate COTS hardware and software components and tools from different engineering fields so they can work together efficiently. The design methodology of our COTS-based real-time control system follows a versatile integrative design approach and architectural framework (VIDAF) [32], which supports the development and implementation of reliable real-time control systems and applications using commercial off-the-shelf components. The design process utilizes real-time techniques and open-source components that support the development of COTS-based real-time control applications. As illustrated in Figure 2, the design and development process is carried out in parallel with the definition of the system requirements and specifications, since all may influence each other (e.g., changes to requirements).

Some of the techniques utilized in this methodology, particularly in embedded control with microcontroller boards, build upon previous research [33,34]. Techniques, such as model-based representations and component-based design methodology [35], were applied together with real-time multithreaded scheduling techniques [36] in designing the real-time control system according to system's specification requirements. The methodology addresses issues and challenges, such as availability and reliability, that pose the selection and integration of COTS components into the development of real-time control system applications.

The overall design and development approach involves the initial requirements and specifications of the real-time system, their applicability analysis, the COTS hardware and software modules architecture (COTS nodes), their model-based simulation verification, and the actual implementation of the developed control system. The requirements analysis determines the design specification parameters that the COTS components need to satisfy according to the needs of the real-time application. Next, the design proceeds with the appropriate integration of COTS components into the

control system architecture. The system components can be either hardware (HW) or software (SW) units. Prior to this stage, appropriate selection of existing or the design of new COTS HW and SW components (e.g., open-source hardware and software), that most fulfill the real-time specifications and requirements, is a necessity.

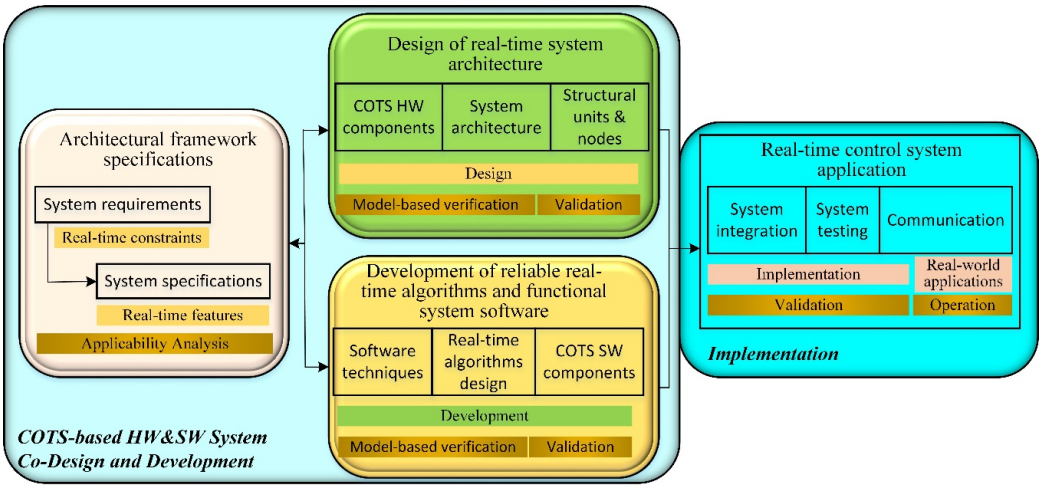

Figure 2. Overview of the design methodology.

The concurrent design of system hardware and software components leads to shorter development times and lower cost. A model-based methodology for verification purposes and diagnosis is based upon a qualitative modeling and simulation tool (QMTOOL) [37]. An important confirmation step is whether the system model fulfills all the requirements. For this purpose, its performance was verified through lengthy simulations, although there is no way to 100% verify the credibility of a model.

Thereafter, based on a reliable system model, a prototype control system is realized. This prototype system is experimentally tested and validated until its performance is acceptable and reliable, prior to its implementation. In this way, this methodology applies simulation-based verification techniques at an early design stage and real-world validations through experimentations with the actual machine, to deliver efficient implementation of such a system at reduced cost and time.

Open-source software and hardware that use an open development license and the use of general-purpose components are some of the key features of the design approach followed. Such components are low-cost and immediately available off-the-shelf. The functionality of a COTS-based component could also be upgraded and extended via custom development. Therefore, the use of readily available COTS hardware/software components is an additional competitive advantage over the use of custom-built components.

2.3. COTS-Based Real-Time Control Architecture

The architectural design of the real-time control system should meet specific functional requirements and quality attributes. Architectural functional constraints, such as timing constraints and performance requirements, should be imposed on the control architecture to ensure the required functionality. In VIDAF framework architecture, the fundamental structures are functional nodes (COTS nodes), which represent high-level objects consisting of hardware and software components, tasks, and properties. Therefore, a COTS-based real-time control system application is structured using such individual functional nodes. Each COTS node may have its own operating system and hardware structure. A run-time middleware layer (Linux firmware with PREEMPT_RT real-time support), that lies between the hardware components of the COTS nodes and the applications running on them, provides the software services and nodes communication support necessary for real-time application development. Furthermore, this results in lower execution latencies and increased number

of execution iterations of the real-time applications running. This is because a real-time task during its execution reacts in time to events that take place, within the desired real-time constraints. Therefore, the choice of preemptive Linux (with real-time support) increases the determinism of the execution of tasks (processes/threads) and reduces the worst-case response times.

Following this architectural approach, in our design, the application's control tasks are multithreaded processes. Each task may have its own real-time scheduling policy and resources, and may run within a single or several distributed COTS nodes. Tasks from different applications may run on different nodes in parallel and even communicate to process the same data. Such a mode of operation may increase bus contention and threads memory interference and cause additional delays in execution of tasks. Therefore, it is preferable to assign and execute tasks only in individual COTS nodes. It is proved that such application infrastructure improves the determinism of a real-time task and keeps the overall system response time within specific real-time bounds [32].

Upon this approach, the development of a COTS-based application system in pump motor control is selected for demonstration and validation purposes. This study aimed to investigate the feasibility of this particular control system and to further explore key features, such as the reliability and effectiveness, of COTS system components.

2.4. Performance Analysis Metrics

The performance evaluation analysis intends to determine the efficiency of the proposed COTS-based control system in the context of its intended use. The analysis criteria address performance metrics, such as functionality, reliability, availability, and cost. Particular attention is given on investigating the feasibility of implementing a reliable and functional controller based on commercial off-the-shelf hardware and software components and open source software. The capability of the system to perform appropriately within specific timing constraints (e.g., controller's response time) as required for a real-time control system is considered in our performance analysis, too.

3. Results

3.1. Case Study: Control of Suction Pump Induction Motor

We investigated the feasibility of using COTS components to real-time control and the feasibility of applying the proposed control system architecture, through a case study in machine construction. This refers to the real-time control of a suction pump motor of a lime milk machine in ADAMs Machines Constructions Company. The company is a family-owned and operated business that designs and builds high-quality machines and support equipment, such as industrial presses, concrete mixers, molds, conveyors, etc., for construction companies and the concrete products industry.

3.1.1. The Pumping System under Control

The pumping system under control is a substantial part of a lime milk machine (Figure 3). The machine is equipped with a lime storage tank and utilizes a suction pump to distribute the lime milk (calcium hydroxide or hydrated lime) to distant storage places and containers. The machine stores and transfers lime milk from the lime mixing machine either to distant, or lime storage pits with significant altitude difference. The suction pump is a single worm screw pump, a product from CMO POMPE S.N.C., Orgiano (VI), Italy, of type PS series [38]. This type covers a flow rate that ranges from 50 to 4000 L/min and is driven by an EDK induction motor (380 V, 7.5 KW, 10 HP, 2500 rpm, 50 Hz). Three-phase induction motors (also known as asynchronous motors) are the most common motors used for industrial machinery, such as hydraulic presses, concrete mixers, pumps, conveyors, etc., because they are self-starting, powerful, and efficient. Such motors have simple fixed winding (stator) and rotor. Therefore, they are cheaper, simpler, and more reliable. VFD (Variable Frequency Drive) motor control is the most efficient and widespread speed control technology with AC induction motors. A variable frequency drive controls the speed of a motor by varying the AC input frequency.

Initially, this is done by converting the existing fixed frequency and voltage AC supply input to direct current (DC) using a rectifier and then converting it to a variable frequency and voltage output using an inverter. Since the inverter is the key component of this control scheme, a VFD is thus quite often simply called an inverter.

Figure 3. Lime milk storage and delivery machine.

A custom-built controller developed within the company, regulates the motor's supply frequency and operates the pump. The custom-design of the pump's motor controller was based upon a MC68705R3 Motorola single chip microcontroller, a member of the MC6805 family of microcontrollers [39]. This microcontroller unit comprises an 8-bit CPU with 112 bytes of RAM, 4K bytes of Erasable Programmable Read-Only Memory (EPROM), four 8-bit I/O ports, a timer, and a four channel A/D converter. The block diagram of this MC68705R3-based custom-built controller is given in Figure 4.

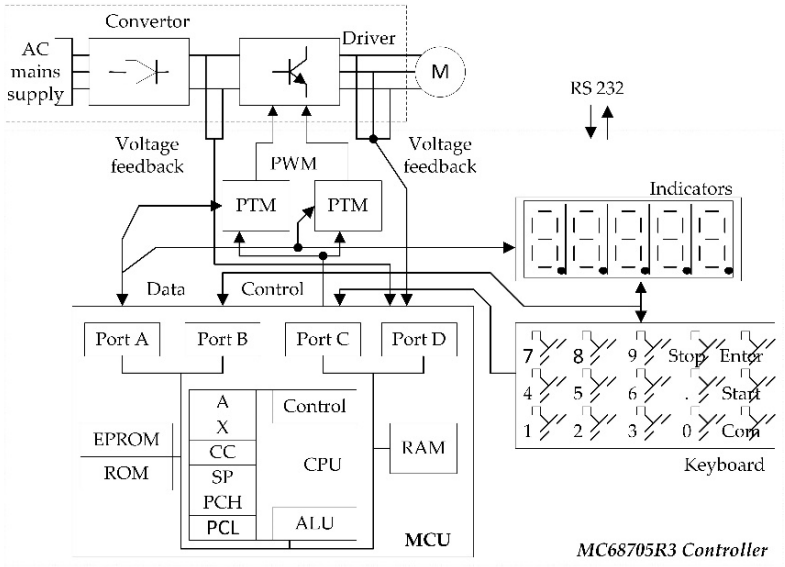

Figure 4. MC68705R3-based custom-built controller.

The converter (or rectifier) converts the given variable voltage into fixed voltage, using a chock inductor and a condensator (dc link). The DC link uses reservoir capacitors to provide a steady DC supply. The driver scheme (three-phase inverter) is based on a PWM inverter built with six power transistors. The single chip MC68705R3 microcontroller has three programmable timer modules MC6840, two of which are being utilized to drive the inverter by the generation of pulse width modulation signals. This is accomplished through lookup tables of time values stored in the microcontroller's memory that correspond to the waveform's frequency values. The application software was written in assembly language using the MC68705R3 CM65x Cross Macro Assembler facilities. An attached keyboard provides the means for input of regulation commands.

3.1.2. The BeagleBone Black-Based Controller

The objective was to build a low-cost COTS-based controller, easily customizable, and more reliable. For this purpose, a BeagleBone Black microcontroller board is selected as the development platform. The BBB microcontroller integrates an ARMv7 Cortex-A8 processor unit within a TI Sitara AM3358AZCZ100 SoC (System-on-Chip), running at 1GHz, with 512MB DDR3 of RAM and 4GB 8-bit embedded Multi-Media Card (eMMC) on-board flash storage. It provides onboard support for 7 ADC (analog-to-digital-converter) channels, 65 GPIOs (general-purpose I/O pins), 4 timers, and 8 PWM outputs. Some of the factors that imposed the selection of this microcontroller include its wide use for embedded low-powered control applications, its computing power, comparative price, and real-time support. The SoC contains two Programmable Real-Time Units (PRUs) programmed in assembly or C, which have access to BeagleBone's RAM, GPIO pins, 32-bit timers, counters, and other SoC's peripherals. Research also indicates that such a device is capable of supporting sufficiently real-time techniques for timely and accurate measurements [40].

The BeagleBone Black unit comes with Debian Linux (Linux BeagleBone 3.8.13-bone70) pre-loaded on the board's flash. In order to provide a specific operating platform, UBUNTU 18.04 LTS Linux was installed and patched with PREEMPT_RT for real-time processing. Linux patched with PREEMPT_RT provides the appropriate firmware (kernel 4.14.74-rt44-v7) for the realization of the control and execution of the real-time applications of the embedded control system [41]. In a real-time system with hard timing constraints, it is important that all timing constraints are fulfilled. Linux OS with PREEMPT_RT real-time support has a preemptive kernel, which is an essential necessity for time sensitive embedded control systems. This feature is important for any real-time application that needs, e.g., to read data from various sensors and respond on time. Certainly, the BBB's ARM Cortex-A8 processor running at 1 GHz is capable to execute sufficiently fast enough nearly 100,000 instruction cycles in 100 μs, which means that a fully preemptive Linux OS kernel might not be necessary. However, it is not easy to determine its real-time behavior, particularly unbounded latencies, when hard real-time constraints are imposed by the system application.

The replacement of this custom-built controller with a new COTS-based controller is the focus of this implementation. This new BBB-based control architecture is shown in Figure 5 and the experimental setup in Figure 6. This control scheme basically features: the BeagleBone Black COTS component, the optocouplers (STMicroelectronics BTA08-600CW3G), and triac circuits (Fairchild Semiconductor Co MOC3061) modules.

3.1.3. PWM Control

PWM techniques are commonly used for controlling the inverter. In the induction motor, the rotor speed varies depending on the strength of the magnetic field. Varying the voltage, along with the frequency (a method preferred in open loop applications), we affect the strength of the magnetic field and the rotor turning, so that we control the motor speed. The ratio V/Hz is kept constant in order to avoid the variation in the magnetic field. Therefore, speed control of the induction motor is achieved through frequency regulation within the range of 20 Hz to 60 Hz.

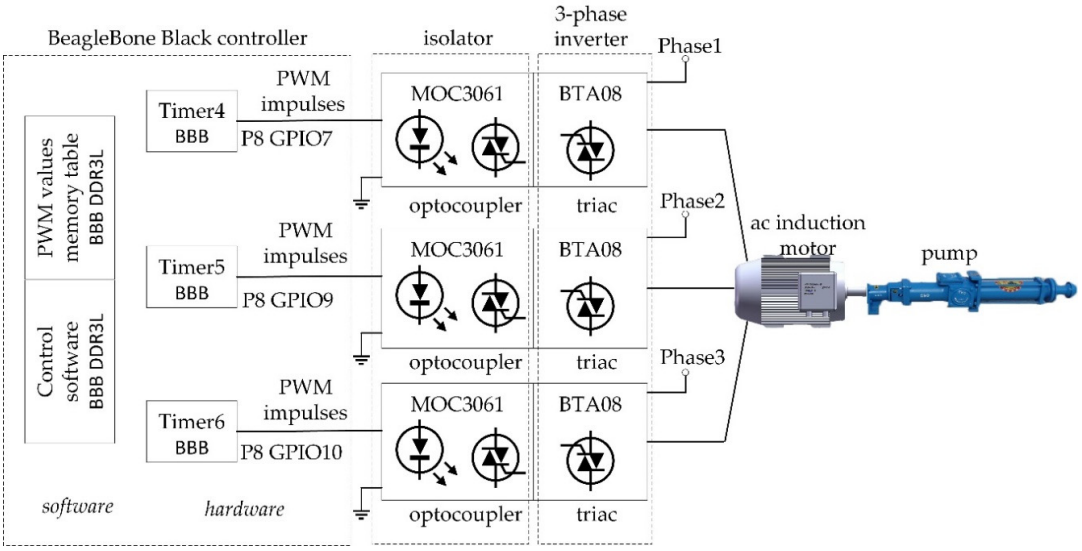

Figure 5. BeagleBone Black-based control architecture.

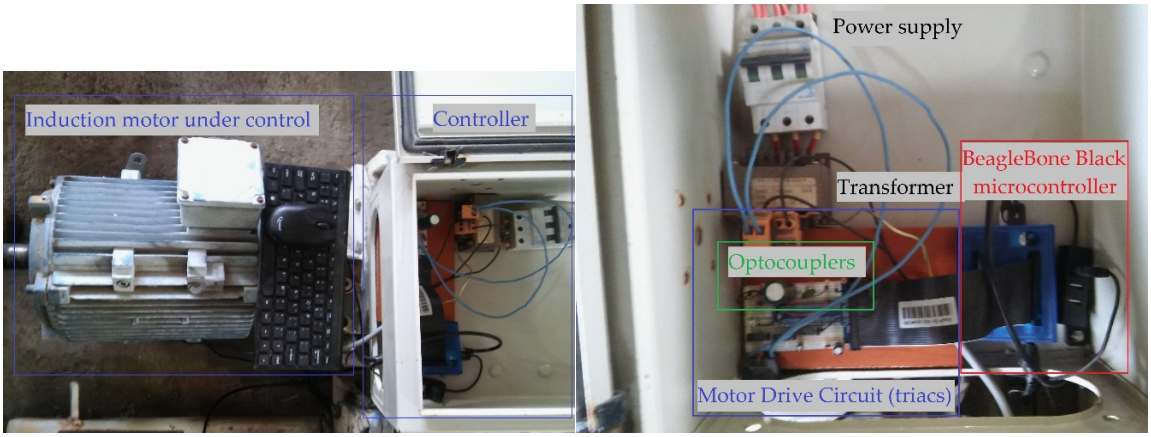

Figure 6. The experimental setup of the controller.

In the three-phase induction motor, stator currents are sinusoidal and form a symmetrical three-phase electrical scheme, due to the electrical displacement of 120° between the three-phase windings. Therefore, it is essential to generate three sinusoidal current waveforms displaced by 120° to control the motor's rotor turning speed. For this purpose, the control scheme utilizes three built-in timers (timers 4, 5, and 6 linked to BeagleBone's P8 header GPIO pins) to generate the pulse width modulation signals that modulate the three phases of the main's supply with PWM impulses. This is achieved through lookup tables of values stored in the controller's memory, representing the sinusoidal waveforms. The timers operate in single impulse generation mode and control the switching time (ONs and OFFss) of the triacs's transistor keys that drive the motor. This process is realized in software.

The system software was developed under the Cortex-A8 development environment and written in C language and partially in ARM assembly. The software computes the PWM values modulating the three-phase AC supply. These values are stored as memory arrays, forming the sine waves that correspond to frequencies ranging from 20 Hz to 60 Hz. The software counts the duration of each phase (e.g., positive half cycle of the modulation signal) and modulates it with the same polarity and amount (eighteen) of PWM impulses. The modulation period may vary with each waveform, which depends upon the frequency of the waveform. The PWM period of each impulse varies between 1851 μs and 5555 μs, respectively, for frequencies of 60 Hz and 20 Hz. Some indicative sets of frequencies, modulation period, and actual speed achieved are shown in Table 1. Results show that the actual

motor speed varies according to modulation period and increases appropriately with the increase of the input frequency.

Table 1. PWM frequency control of motor's operation.

Set	Set Frequency (Hz)	Modulation Period (μs)	Actual Speed (rpm)
1	20	5555	755
2	30	4166	947
3	40	2777	1138
4	50	2314	1710
5	60	1851	2280

The flowchart of the master control program for PWM generation is shown in Figure 7. Initially, system variables, timers, and counters are set and initialized. Three BBB's timers control the output width of the PWM signals and memory counters count the parts of every half cycle (positive and negative semi-periods) of the three sinusoidal signals. Each half of a cycle (semi-period) is divided in nine parts (steps). At time intervals equal to 1/9 of the time period of the sine wave, a timer produces an interrupt so that the PWM values are loaded for the corresponding frequency.

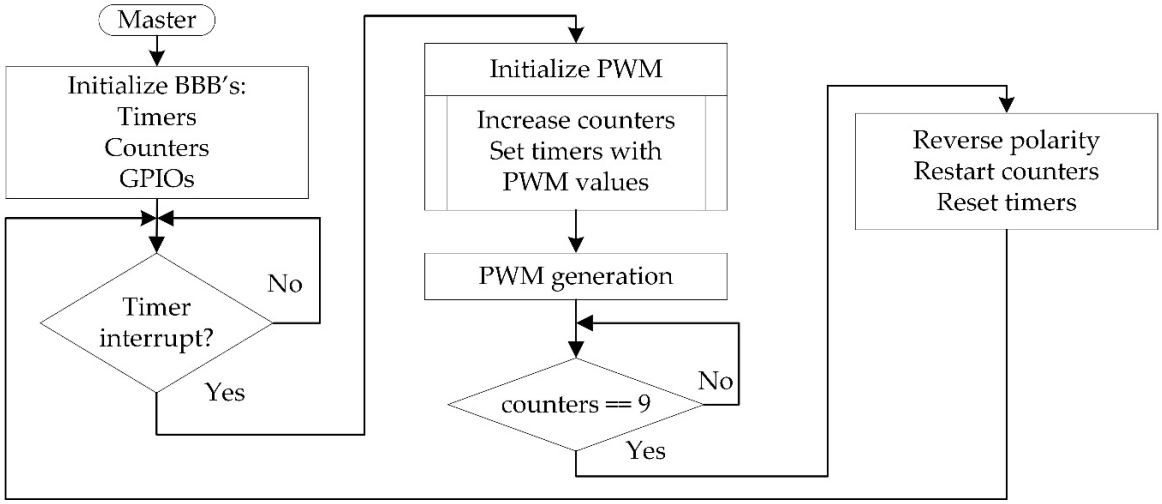

Figure 7. Flowchart of the master pulse width modulated (PWM) generation control program.

Each PWM value uses an 8-bit step size. This yields an amount of eighteen bytes for the PWM values that modulate the entire sine wave. During the modulation, each of the three timers is set with the PWM values for each part of the sine wave. This setting is done accordingly to the counters' contents that trace the polarity change of each sine wave. When a counter's value equals to nine (half cycle), then the polarity of the signal is reversed, the counter is restarted, and the timer is reset. Otherwise, the timer is set to a value equal to this frequency step and continues the functional cycle. The amount of steps is predetermined and equal to 9, but the time length of each step is different for each frequency. In low frequencies, it is greater, while it is smaller in high frequencies. This realizes the continuous modulation of sine waves generation that yields to speed regulation and control of motor's operation.

Overall, the control loop is supposed to be stable; however, potential critical conditions could have been further investigated. For this purpose, future work will investigate this issues by using further techniques, such as model predictive control, based on the work of Mercorelli et al. [42].

3.1.4. Simple Control Interface

A top layer of the PWM control software is a simple graphical user interface (GUI), shown in Figure 8, used to facilitate the experiments.

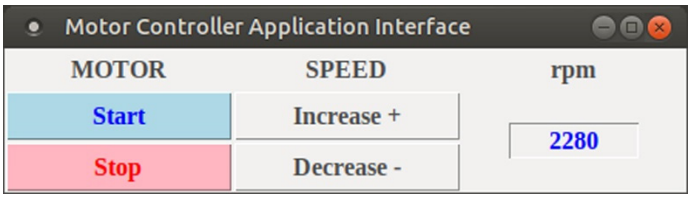

Figure 8. Motor controller application interface.

The interface enables easy access to microcontroller's I/Os controlling the motor's speed. The principle of control behind the interface involves sending commands to the PWM control software according to increase/decrease of the modulation period of the PWM frequency and duty cycle. Simple built-in functions in Python communicate with the C functions in the PWM master control module which apply the PWM variations in frequency (increase/decrease the modulation period). The actual speed (rpm) of the motor is estimated computationally, based on the set frequency and the modulation period, instead of being measured. This is due to the fact that the control is open loop and focuses mainly on the actual PWM control of motor's input without feedback monitoring. However, the maximum actual speed was measured for verification purposes using a tachometer mounted on the motor's rotor shaft and found to be approximately 2880 rpm.

The graphical user interface (GUI) is created using the Tkinter module, which comes with Python by default. This module provides an interface to Tk GUI toolkit, developed in Tool Command Language (TCL) [43]. This graphics module uses ctypes foreign function library and objects to capture any user-initiated input event (e.g., start/stop motor) and then communicates with the master PWM control module in C to execute the required C functions of operation. These C functions are defined in a shared library that is created as a shared object that the Python GUI module executes.

3.2. Performance Analysis

A set of experiments was carried out to test the functionality and operation, and any improvements on performance, of the new BeagleBone Black-based controller. The testing conditions, such as reference voltage, frequency, and reference load, remain the same in both controllers. The controllers' performance was investigated under different testing scenarios, which involve variation in speed and applied load. As it is shown in Table 2, the experimental results and analysis show that the time (settling time) taken by the motor to reach the reference speed of 2300 rpm (actual speed is 2280 rpm) under no load is 0.52 s at the old controller and 0.31 s for the new controller. When the rated load on the pump motor is increased to 100% of the full supply, the time taken by the motor to reach the reference speed of 2300 rpm (actual speed is 2280 rpm) is 1.44 s for the old controller and 1.10 s for the new controller. The BBB controller's reaction time in real-time is about 10 ms quicker than the old MC68705R3-based controller. This analysis shows that the new controller provides better performance at an average rate of +32% compared to the old controller. In addition, the BeagleBone Black ARM-based embedded controller improves some of the control systems' quality criteria, e.g., its immediate availability and reusability of hardware and software components whenever system requirements are changed. The MC68705R3-based controller is hard to be re-programmed (in low-level assembly language). In addition, its 8-bit hardware structure is difficult to support further peripheral control devices (sensors and actuators). Therefore, it could not easily be reused for further control operations of the machine under control. On the other hand, the BBB-based COTS controller, which integrates a 32-bit ARM CPU running a real-time Linux OS, could easily be adapted to hardware or software changes

and new technologies and, therefore, maximize the reuse of its software and hardware components and functionalities to support new control needs.

Table 2. Experimental measurements of the two controllers.

Parameters	Load (%)	Reference Speed (rpm)	Settling Time (s)	Controller's Reaction Time (ms)	Performance Improvement (%)	Software and/or Hardware Reusability
MC68705R3 controller	no load 100%	2300 (actual 2280)	0.52 1.44	12	- -	no
BeagleBone controller	no load 100%	2300 (actual 2280)	0.31 1.10	2	+40% +24%	yes

Although several aspects have an impact on the overall costs for a product, these include mainly purchasing and development costs, operation testing and verification costs, implementation or system integration costs (wiring of sensors and actuators, etc.), certification costs (e.g., ISO, CE required for functional safety applications), and, finally, maintenance costs.

In the case of the MC68705R3-based custom-made controller, the development costs include extra manufacturing costs for the specific PCB (printed circuit board) controller design, time-consuming efforts in developing the low-level control software, and extended testing and verification procedures with specific test electronics (circuit and functional tester). Implementation and certification costs are kept high due to extensive and expensive company prototyping procedures until the desired functionality of the final product is reached. Maintenance and operation costs typically are not significant. However, in some cases of change or expansion in the functionality of the machine's operations, its low-level control software could not easily be adapted.

On the other hand, the development costs for the BBB-based COTS controller are lower. This is due to the following: there is no need of any specific manufacturing, the development of the control software primarily in C high-level programming is easier, less time-consuming, and therefore less expensive, and it requires fewer testing and verification procedures. Implementation and certification costs are lower due to plug and play capability of the BBB controller and its ease of connection to several peripheral devices. The controller's already specified certifications for certain applications (typical for COTS devices as the BeagleBone Black to have some pre-certification) could ease future certification procedures. The costs for maintenance and potential expansion or adaptation to (hardware/software) changes are not significant. The BBB-based COTS controller could easily adapt to potential replacement of peripherals or other control modules, and monitoring features could easily be implemented via an application interface developed in Python for this purpose.

Experiments were carried out with the actual lime milk machine. During all the experiments, the operating conditions and lime milk attributes (e.g., lime tank pressure and quantity, lime milk volume, lime temperature, etc.) were kept constant and the same for both control systems. Both controllers are low-powered (3–5 V) and energy-efficient. Measurements (with a single-phase power analyzer) show that their differences in energy savings are negligible, as both controllers consume power less than 2 W in a working period of 12 h.

4. Discussion

Despite the architectural differences between the two controllers, the experimental results show that both control systems render reliable results that meet the operational criteria set. In particular, the BBB's PWM controller performs quite efficiently in providing stable motor speed regulation, as well as shows improvement of the operation of the system under control, compared to the previous MC68705R3-based controller. The comparative analysis of the results of the two controllers reveals that the new BeagleBone Black-based controller has better reaction time, reduces the motor's speed settling time, and improves some of the system's quality features. Overall, this controller proves to be efficient in qualitative terms, such as availability, versatility, and reusability (see Table 3). In general, using COTS

components that can be ordered immediately from various vendors speeds up the development since we have less to build and facilitates the development of an application with high-level software and programming tools. However, vendors may not provide design information and may impose restrictions on their intellectual property. The amount of reusable software (e.g., Linux OS kernel modules, C code, and libraries) and hardware (e.g., the BBB microcontroller) is quite considerable, particularly since the same microcontroller could be utilized for other machine control functions, too. These findings might be useful for researchers who are interested in experimenting with general-purpose microcontroller boards, such as BeagleBone Black, into the control of induction motors.

Table 3. Qualitative metrics comparison of the two controllers.

	Programming Model	**Runtime**	**Availability**	**Versatility**	**Reusability**
MC68705R3 controller	Motorola assembly	No OS	Custom order	Specific-purpose	Low
BeagleBone controller	C and ARM assembly	OS with real-time kernel	Immediate	General-purpose	High

The MC68705R3-based custom-made controller was developed from the scratch and this required a significant effort and a considerable cost, particularly in the development of the control software. As a result, it has an optimized functionality tailored for the system under control. However, its development, maintenance and potential expansion costs overall are not so attractive. Additionally, functional changes in the operation of the system under control leads to serious reconsideration of the assembly control software or even its hardware components.

On the other hand, the BBB-based COTS controller has lower development and implementation costs. Additionally, based on a high-level real-time Linux OS could easily adapt to functional and structural changes often by applying a plug and play procedure or high-level re-programming. Especially in cases of critical-safety applications, its real-time features and functional reliability could have a positive impact on the safety properties of the overall system under control and could assure the fulfillment of specific functional, timing, or other quality constraints (e.g., scope, cost, etc.). Of course, although the BBB-based COTS controller was not applied in a critical-safety application, it has the capacity to support such needs, especially for real-time applications.

The proposed COTS-based embedded control system demonstrates the ability to perform PWM frequency control of a pump's induction motor function. In our experience, this COTS-based approach takes significantly less effort than the custom-based approach, saving costs and time the major concerns for the company in building the machine. In addition, this implementation is generally more adaptable to change and modifications due to the use of readily available and up-to-date commercial off-the-shelf components. However, in some cases, the integration of COTS components may place some constraints on the rest of the system design (e.g., in the use of specific peripheral equipment) or even impose some difficulties in finding suitable replacements if the COTS vendor stops manufacturing the COTS component.

Since this new controller is an incremental improvement to the existing custom-built one, a prototype was required to test and verify key its performance. Although it is an experimental prototype, it fulfills its intended purpose and offers specific advantages compared to previous implementation and approach. Some of the advantages can be identified below:

- Open-source, adaptable control system, based on a general-purpose ARM-based BeagleBone Black microcontroller.
- Short-time application development and easily modifiable code based on C high-level language.
- Use of Linux OS with PREEMPT_RT real-time support, combined with the BeagleBone's programmable real-time units, is capable to support performance enhancements according to future real-time control requirements.

The selection of the BeagleBone Black microcontroller board with programmable real-time units support and the open source Linux OS with real-time support have significant value from the control

point of view since both could be reconfigured to extend the applicability of the proposed control system to other target applications. For any control application, it is important that the controller can be adapted to changing requirements as it is the case of the BBB controller. This type of controller is well suited for applications where experimentation with low-cost real-time hardware and software components is of primary importance.

The research outcomes are expected to benefit the design and development methodologies of such systems and, consequently, their deployment in real-time applications. The proposed methodology and research approach could be applicable in a wider set of domains. This could enhance the approaches in building reliable real-time control systems based on general-purpose COTS hardware/software components.

5. Conclusions

The choice of COTS hardware and software components has additive value for the low-cost development and maintenance of control systems. However, their reliability and efficiency in real-time control applications is still under research. This research applies a versatile open source approach that uses commercial off-the-shelf (COTS) components in the development and implementation of a novel real-time control system for a pump motor. The experimental results validate the feasibility and applicability of the proposed approach in a real-world case study, as well as justify the efficiency of the proposed control architecture compared to previous control implementation.

An emphasis of this research is placed on finding efficient system solutions for real-time applications based on commercially available off-the-shelf hardware/software components. Therefore, the performance of the new ARM-based BeagleBone Black control scheme was investigated in comparison to the MC68705R3-based microcontroller. Performance analysis shows that the BeagleBone Black COTS-based controller provides better performance and improves some of the control systems' quality criteria. The research also tackle issues, such as the reliability and efficiency of real-time control systems, based on COTS components. Another outcome of this research is that such a COTS-based approach can reduce the development time and ease the implementation of low-cost experimental testbed systems for further research in real-time control based on COTS components. Future work could focus on providing a more compact version of the controller, with improved features, such as wireless control and monitoring, for further research and exploitation purposes.

Author Contributions: Conceptualization, G.K.A.; Methodology, G.K.A. and N.P.; Software, G.K.A.; Validation, P.A.K. and T.S.; Investigation, G.K.A.; Data curation, G.K.A., N.P., and P.A.K.; Writing—original draft preparation, G.K.A.; Writing—review and editing, G.K.A., N.P., P.A.K., and T.S.; Visualization, P.A.K. and T.S.; Supervision, G.K.A. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by ADAMs Machines Constructions Company, grant number 125.

Acknowledgments: The authors want to acknowledge the ADAMs Machines Constructions Company for providing the infrastructure, technical help and assistance in carrying out this research.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sathishkumar, S.; Meenakumari, R.; Jobanarubi, E.; Anitta, P.J.S.; Ravikumar, P. Microcontroller Based BLDC Motor Drive for Commercial Applications. In *Power Electronics and Renewable Energy Systems. Lecture Notes in Electrical Engineering*; Kamalakannan, C., Suresh, L., Dash, S., Panigrahi, B., Eds.; Springer: New Delhi, India, 2015; Volume 326, pp. 829–841. [CrossRef]
2. Tomei, P.; Verrelli, M.C.; Marino, R. *Induction Motor Control Design*; Springer: London, UK, 2010.
3. Jamadar, B.N.; Kumbhar, S.R.; Gavane, P.M.; Sutrave, D.S. Design and Development of Control System for Three Phase Induction Motor using PIC Microcontroller. In Proceedings of the 3rd International Conference on Advances in Control and Optimization of Dynamical Systems, Kanpur, India, 13–15 March 2014; pp. 807–811. [CrossRef]

4. Kamel, K.; Kamel, E. *Programmable Logic Controllers: Industrial Control*; McGraw-Hill Professional: New York, NY, USA, 2013.
5. Giri, F. *AC Electric Motors Control: Advanced Design Techniques and Applications*; John Wiley & Sons, Ltd.: Toronto, ON, Canada, 2013. [CrossRef]
6. Rashid, H.M. *Power Electronics Handbook*, 4th ed.; Elsevier Butterworth-Heinemann: Oxford, UK, 2018. [CrossRef]
7. Lu, X.; Chen, S.; Wu, C.; Li, M. The Pulse Width Modulation and its Use in Induction Motor Speed Control. In Proceedings of the Fourth International Symposium on Computational Intelligence and Design, Hangzhou, China, 28–30 October 2011; pp. 195–198. [CrossRef]
8. Jat, G.L.; Singh, K.; Mahajan, A.; Shimi, S.L. Real time speed control of induction motor using new generation DSP controller. In Proceedings of the 2016 Second International Innovative Applications of Computational Intelligence on Power, Energy and Controls with their Impact on Humanity (CIPECH), Ghaziabad, India, 18–19 November 2016; pp. 90–95. [CrossRef]
9. Menghal, P.; Laxmi, A.J. Real time control of electrical machine and drives: A review. *Int. J. Adv. Eng. Technol.* **2011**, *1*, 112–126.
10. Quintero-Manriquez, E.; Sanchez, E.N.; Felix, R.A. Real-Time Direct Field-Oriented Control of induction motor for electric vehicles applications. In Proceedings of the 2014 IEEE International Autumn Meeting on Power, Electronics and Computing (ROPEC), Ixtapa, Mexico, 5–7 November 2014; pp. 1–6. [CrossRef]
11. Pulle, D.W.J.; Darnell, P.; Veltman, A. *Applied Control of Electrical Drives*; Springer International Publishing: Cham, Switzerland, 2015. [CrossRef]
12. Prasad, S.A.H.; Kariyappa, B.S.; Nagaraj, R.; Thakur, S.K. A novel technique of AC power control using micro controller based inverter. In Proceedings of the 9th International Conference on Signal Processing, Beijing, China, 26–29 October 2008; pp. 2713–2716. [CrossRef]
13. Jannati, M.; Anbaran, S.A.; Asgari, S.H.; Goh, W.Y.; Monadi, A.; Aziz, M.J.A.; Idris, N.R.N. A review on Variable Speed Control techniques for efficient control of Single-Phase Induction Motors: Evolution, classification, comparison. *Renew. Sustain. Energy Rev.* **2017**, *75*, 1306–1319. [CrossRef]
14. Mikhael, H.; Jalil, H.; Ibrahim, I. Speed Control of Induction Motor using PI and V/F Scalar Vector Controllers. *Int. J. Comput. Appl.* **2016**, *151*, 36–43. [CrossRef]
15. Aspalli, M. *Microcontroller Based Controller for Three Phase Induction Motor*; LAP LAMBERT Academic Publishing: Riga, Latvia, 2020.
16. Datta, S.; Chandra, A.; Chowdhuri, S. Design and development of an 8 bit microcontroller based space vector PWM inverter fed volt/Hz induction motor drive. In Proceedings of the 2nd International Conference on Control, Instrumentation, Energy & Communication (CIEC), Kolkata, India, 28–30 January 2016; pp. 353–357. [CrossRef]
17. Samman, F.A.; Waris, T.; Anugerah, T.D.; Mide, M.N.Z. Three-phase inverter using microcontroller for speed control application on induction motor. In Proceedings of the Makassar International Conference on Electrical Engineering and Informatics (MICEEI), Makassar, Indonesia, 26–30 November 2014; pp. 28–32. [CrossRef]
18. Payak, M.; Kumbhar, S.R. FPGA based PWM control of induction motor drive and its parameter estimation. In Proceedings of the International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), Davangere, India, 29–31 October 2015; pp. 631–635. [CrossRef]
19. Romlie, M.F.B.; Pesol, M.F.; Hasan, K.N.M. PWM technique to control speed of induction motor using Matlab/xPC target box. In Proceedings of the IEEE 2nd International Power and Energy Conference, Johor Bahru, Malaysia, 1–3 December 2008; pp. 718–721. [CrossRef]
20. Alwadie, A. The Decision making System for Condition Monitoring of Induction Motors Based on Vector Control Model. *Machines* **2017**, *5*, 27. [CrossRef]
21. Hur, B. *Learning Embedded Systems with MSP432 Microcontrollers: MSP432P401R with Code Composer Studio*, 2nd ed.; Independently Published: College Station, TX, USA, 2020.
22. Mohamed, A.; Ruhe, G.; Eberlein, A. COTS selection: Past, present, and future. In Proceedings of the 14th Annual IEEE International Conference and Workshop on Engineering of Computer Based Systems (ECBS), Tucson, AZ, USA, 26–29 March 2007; pp. 103–114. [CrossRef]
23. Salewski, F. COTS or Custom Made? Design Decisions for Industrial Control Systems. In Proceedings of the 8th International Conference on Advances in Circuits, Electronics and Micro-Electronics (CENICS), Venice, Italy, 23–28 August 2015; pp. 7–12.

24. Sethi, A.; Thakurta, V.; Gajanur, N.; Cheela, B.S.; Sadasivan, K.S.; Hosangadi, R. Implementation of COTS components for CubeSat applications. In Proceedings of the IEEE Aerospace Conference (AERO 2017), Big Sky, MT, USA, 4–11 March 2017; pp. 1–11. [CrossRef]
25. Sabripour, S.; Haque, J.; Ciszmar, A.; Magesacher, T. A COTS-based software-defined communication system platform and applications in LEO. In Proceedings of the 36th International Communications Satellite Systems Conference (ICSSC 2018), Niagara Falls, ON, Canada, 15–18 October 2018; pp. 1–5. [CrossRef]
26. Lovascio, A.; D'Orazio, A.; Centonze, V. Characterization of a COTS-Based RF Receiver for Cubesat Applications. *Sensors* **2020**, *20*, 776. [CrossRef] [PubMed]
27. O'Halloran, M.; Hall, J.G.; Rapanotti, L. Safety engineering with COTS components. *Reliab. Eng. Syst. Saf.* **2017**, *160*, 54–66. [CrossRef]
28. Nack, A.M. Standardizing Functional Safety Assessments for Off-the-Shelf Instrumentation and Controls. Master's Thesis, University of Tennessee, Knoxville, TN, USA, May 2016.
29. Butler, E.; Fletcher, G.; George, S.; Guerra, S.; Khlaaf, H. *COTS Digital Devices in Safety Critical Industries*; Energifors: Stockholm, Sweden, 2019; pp. 1–38.
30. Agrawal, V.K.; Agrawal, V.K.; Taylor, R.A. Trends in Commercial-Off-The-Shelf vs. Proprietary Applications. *J. Int. Technol. Inform. Manag.* **2016**, *25*, 1–36.
31. Al-Ali, A.; Zualkernan, I.A.; Rashid, M.; Gupta, R.; AliKarar, M. A smart home energy management system using IoT and big data analytics approach. *IEEE Trans. Consum. Electron.* **2017**, *63*, 426–434. [CrossRef]
32. Adam, G.K.; Petrellis, N.; Garani, G.; Stylianos, T. COTS-Based Architectural Framework for Reliable Real-Time Control Applications in Manufacturing. *Appl. Sci.* **2020**, *10*, 3228. [CrossRef]
33. Adam, G.K.; Kontaxis, P.A.; Bouroussis, C.A.; Ventzas, D.E.; Topalis, F.V. Embedded computer communication and control of DALI LED drivers. In Proceedings of the Balkan Light Conference, Athens, Greece, 16–19 September 2015; pp. 125–130.
34. Adam, G.K.; Kontaxis, P.A.; Doulos, L.T.; Madias, E.-N.; Bouroussis, C.A.; Topalis, F. Embedded Microcontroller with a CCD Camera as a Digital Lighting Control System. *Electronics* **2019**, *8*, 33. [CrossRef]
35. Adam, G.K.; Karapoulios, C. Model-based qualitative studies of complex manufacturing systems. *WSEAS Trans. Inf. Sci. Appl.* **2004**, *1*, 88–93.
36. Adam, G.K. Process scheduling evaluations using multithreaded software modules. *AMSE Adv. Model.* **2005**, *10*, 13–26.
37. Adam, G.K.; Grant, E.; Adam, K.G. Qualitative Modelling and Control of Industrial Processes. In Proceedings of the IASTED International Conference on Modelling and Simulation, Pittsburgh, PA, USA, 15–17 May 2000; pp. 477–482.
38. CMO Pompe. General Production Catalogue. Available online: www.cmo-pompe.it/wp-content/uploads/2017/04/cmopompe_catalogo_ita_ing_2017.pdf (accessed on 10 June 2020).
39. Koshal, D. *Manufacturing Engineer's Reference Book*; Butterworth-Heinemann: Oxford, UK, 1993.
40. Alanwar, A.; Anwar, F.; Zhang, Y.-F.; Pearson, J.; Hespanha, J.; Srivastava, M. Cyclops: PRU Programming Framework for Precise Timing Applications. In Proceedings of the International IEEE Symposium on Precision Clock Synchronization for Measurement, Control and Communication (ISPCS), Monterey, CA, USA, 27 August–1 September 2017; pp. 1–6. [CrossRef]
41. The Linux Foundation. Real Time Linux. Available online: https://www.linuxfoundation.org/ (accessed on 29 August 2020).
42. Mercorelli, P.; Kubasiak, N.; Liu, S. Multilevel bridge governor by using model predictive control in wavelet packets for tracking trajectories. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), New Orleans, LA, USA, 26 April–1 May 2004; pp. 4079–4084. [CrossRef]
43. Tkinter-Python Interface to Tcl/Tk. Available online: https://docs.python.org/3/library/tkinter.html (accessed on 22 July 2020).