International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT

ELSEVIER
SSRN

IJIEMR Transactions, online available on 25h Nov 2018. Link
:http://www.ijiemr.org/downloads.php?vol=Volume-07&issue=ISSUE-12

Title: **FPGA IMPLEMENTATION WITH IMPROVED WATCHDOG TIMERS FOR REAL TIME FACILE SYSTEMS**

Volume 07, Issue 12, Pages: 326–331.

Paper Authors

PERLA. SAI BHARGAVI, DUKKIPATI. VENKANNA BABU, DR. JAGAN MOHAN RAO. S

Ramachandra College of Engineering, Eluru

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

FPGA IMPLEMENTATION WITH IMPROVED WATCHDOG TIMERS FOR REAL TIME FACILE SYSTEMS

PERLA. SAI BHARGAVI[1], DUKKIPATI. VENKANNA BABU[2], DR. JAGAN MOHAN RAO. S[3],

[1]M.Tech Student, Dept. Of ECE, Ramachandra College of Engineering, Eluru
[2]Associate Professor, Dept of ECE, Ramachandra College of Engineering, Eluru
[3]Prof. & HoD, Ramachandra College of Engineering, Eluru.
[1]saibhargavi8888@gmail.com, [2]dvenkannababu@gmail.com, [3]jaganmohanrs@gmail.com

ABSTRACT: Embedded systems that are employed in safety critical applications require highest reliability. External watchdog timers are used in such systems to automatically handle and recover from operation time related failures. Most of the available external watchdog timers use additional circuitry to adjust their timeout periods and provide only limited features in terms of their functionality. This paper describes the architecture and design of an improved configurable watchdog timer that can be employed in safety-critical applications. Several fault detection mechanisms are built into the watchdog, which adds to its robustness. The functionality and operations are rather general and it can be used to monitor the operations of any processor based real-time system. This paper also discusses the implementation of the proposed watchdog timer in a Field Programmable Gate Array (FPGA). This allows the design to be easily adaptable to different applications, while reducing the overall system cost. The effectiveness of the proposed watchdog timer to detect and respond to faults is first studied by analysing the simulation results. The design is validated in a real-time hardware by injecting faults through the software while the processor is executing, and conclusions are drawn.

KEY WORDS: watchdog timer; real-time systems; FPGA

I.INTRODUCTION

For applications where a system crash could lead to human injury, highest reliability is required. Such systems should have fault tolerance mechanisms that account for the unexpected to ensure proper safety of operation. These systems should also be able to recover from a crash without any human assistance. These fault tolerance mechanisms detect when a fault occurs in order to handle the fault and to limit the system downtime. One way to achieve fault tolerance is by implementing system redundancy. By using multiple copies of the critical Components of the system, the overall system reliability is enhanced. Parsevals sum of squares checks is the most widely known. In modern communication systems, it is increasingly common. However, this improved system reliability is achieved through increased hardware and software complexity, depending on the type of architecture used. When developing a fault-tolerant system, one of the most cost effective ways of detecting and handling operation time related failures is the watchdog. A watchdog timer (WDT) is a hardware subsystem that monitors the operations of the system and takes certain actions in the event of detecting a fault. It typically consists of a timer circuit and the processor is required to periodically reset the timer. If the WDT expires, it is a secondary indication of some problem

with the system under observation. When the processor fails to reset the watchdog, a decision is made to restart the system or put the system into a known state from which it can recover, thus preventing further damages. A watchdog can be internal (on-chip) or external to the processor. Internal watchdog reduces the hardware complexity and cost, however, is not a robust solution. The software has control over it during runtime and a runaway code can disable the watchdog timer. Moreover, since it is connected to the processor clock, a crystal failure will make the watchdog incapable of monitoring the hardware for faults. When the reliability of an embedded system is crucial, external watchdogs become unavoidable. An external watchdog runs independent of the processor and does not share its clock with the processor. This overcomes the limitations of internal watchdogs and leads to much more robust fault-tolerant system architectures. A class of standalone watchdog timer microchips offer only fixed timeout periods, which make them less generic. Other set of devices allow adjusting the timeout periods by using additional external circuitry. Though useful, this method adds to the complexity of the hardware and increases the overall system cost. The increased cost and complexity of external watchdogs can be managed to a certain extend by realizing the watchdog functionality within a Field Programmable Gate Array (FPGA). Many of the modern embedded systems incorporate one or more FPGA devices to accomplish the desired system functionality. Accommodating the watchdog timer within a FPGA can yield an efficient and robust solution. The work done by Giaconia et al. considered the implementation of a custom concurrent watchdog processor in FPGA for real-time control systems. The design did not provide a timer for the processor; rather, it performed a reasonableness check on some variables and a basic program flow check. El-Attar et al. proposed a sequenced watchdog timer that used time registers to determine whether or not a fault has occurred. However, it did not offer much configuration options and the fault detection features implemented were limited. The authors addressed the basic concepts of a multiple hardware watchdog timer system in FPGA, but kept the design of the watchdog simple.Recent years have seen a growing interest in molecular programming, programming matter at a Nano scale level. Molecular programming draws on computer science and biology to create molecular systems. Instead of using circuits and wires, a molecular system uses structures such as DNA, RNA, or other chemical molecules to perform computational tasks. Molecular systems have been used to create Boolean circuits, perform digital signal processing and build Molecular robots at the Nano scale level. A molecular system can be modelled through the use of chemical reaction networks (CRNs). Stochastic chemical reaction networks (SCRNs) model systems that use finite numbers of molecules through stochastic chemical kinetics. Mass action chemical kinetics can be used to model systems in terms of molecular concentrations, i.e., what percentage of a solution is made up by each species. One way to implement CRNs is to use DNA Strand Displacement (DSD). Molecular systems are inherently probabilistic, having an amount of uncertainty in all

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

interactions. This adds complexity to their construction and use because nothing is guaranteed. SCRNs are a useful programming paradigm. The capabilities of SCRNs have been thoroughly analysed, and they have been compared to a number of other computational models. SCRNs are Turing Universal, able to compute any function a Turing Machine can compute, given an arbitrarily small probability of error.

II. BACKGROUND

A watchdog timer is a device used in safety-critical systems to inform either a user or another system when a specific system fails. A watchdog timer can be as simple as a counter with one input and one output. The counter increments until a specified value and sends a signal on its output to represent an alarm. Upon receiving input, the counter is reset to zero and begins counting again. Nancy Levisohn describes a watchdog timer as a timer that the system must keep restarting. If the system fails to restart the timer within a given period, the watchdog initiates some type of protection action" [20]. John Knight describes a watchdog timer as a countdown timer that is set by an application and which raises an interrupt when it expires". It is a device designed to monitor a single external system and take a specified action, usually issuing an alarm, when the system fails.

Figure 1 and Figure 2 show the two possible outcomes of a watchdog timer where the counter threshold value is set to 4. A watchdog timer has a delay component, an alarm component, and detects the presence of an external heartbeat. In the absence of a heartbeat for some time period, the alarm is triggered. The heartbeat acts to delay the alarm for another time period by resetting the counter. If another heartbeat is not input into the system during the new time period, the alarm will go o_. The heartbeats are added at specified intervals, to keep the alarm from being issued. If the alarm is issued, it means that the monitored system is unable to release a heartbeat and, therefore, is in a failed state.

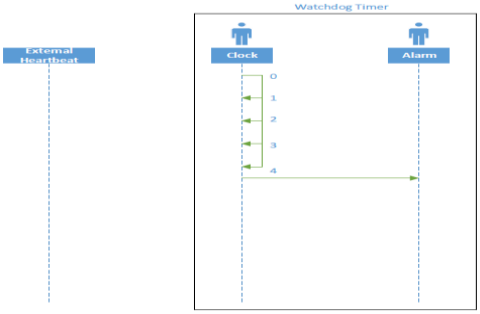

Figure 1 a watchdog timer that does not receive a heartbeat. It counts until reaching its

Specified value and sends out an alarm.

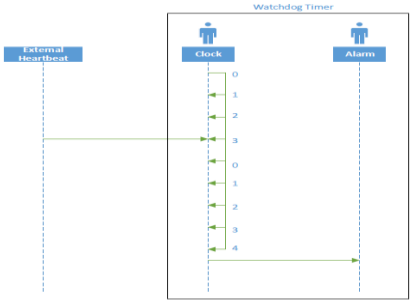

Figure 2 a watchdog timer that receives a heartbeat at 75% of its specified value. **It resets It's counter to zero and begins counting again. It does not receive another heartbeat**

So it releases an alarm upon completion

We used the Goal-oriented Requirements Engineering process denned in to dine our requirements. Van Lamsweerde dense a goal as a prescriptive statement of intent the system should satisfy through cooperation of its agents". A goal models an aspect of the target system in a simple and concise manner. A simple view of the

requirements can be useful for explaining the system to stakeholders. A goal dense a task the system needs to complete. We dine a top-level goal that represents the task. We then renew each goal into sub goals and repeat the process until we have leaf level goals. Goal-Oriented Requirements Engineering distinguishes between behavioural goals and soft goals. Behavioural goals describe intended system behaviour and can be Achieve goals or Maintain/ Avoid goals. An Achieve goal is a target that must be completed. A Maintain goal is a condition that needs to be maintained during operation. An Avoid goal is a condition that needs to be avoided. Soft goals describe preferences between alternative system behaviours and are used to compare alternative options.

III. PROPOSED ARCHITECTURE

Real-time computer systems are defined as systems that are in any conditions able to guarantee their response time. Such systems are used mostly in various embedded devices to guarantee their usability, for example to ensure smooth video playback, and in various industrial control applications. Their utilization in industrial application is often connected with the mission-critical tasks that need to be accomplished in time to prevent system malfunction or damage. The real-time computer system is usually implemented on specific hardware aimed for such purposes. It can run a simple application that takes care of the whole controlled system or an operating system with several applications of which each one has its own task and response deadline defined. One of the methods to recover such systems from error states and ensure their further functionality and responsiveness is utilization of watchdog timers. Watchdog timer is a hardware device usually realized by a counter with match register and specific system connections.The device itself is capable of sending only one message – the indication that the particular timer has overflowed. Each message consists of the opcode byte that defines the operation and the address of the involved timer. The message for loading the timer with new value includes also the value to be loaded into the timer. The control unit has been designed as a sequential circuit without any parallel capabilities. Assuming that the communication with the master device and the timers' service is relatively sparse, this appears to be a sufficient solution. As the control unit also handles the situation when a timer overflows, it has to be notified that such situation has occurred and has to obtain the number of the overflowed timer. The notification of the overflow state is propagated from each timer to the control unit. After the control unit notices this state, it enables the write circuitry to write the number (address) of the overflowed timer to the address bus. This is realized by encoding the actual counter position and placing the encoded address on the address bus. Each of the timers is addressed by its exclusive 8-bit address. When particular timer is accessed, the Serial Load, Communication Clock and Reset inputs are connected to the control unit. Otherwise, the connection with the control unit is disabled and the inputs remain in their default values. All watchdog timers are managed by the control unit that is responsible for their loading, restarting, and reading their overflow flags. It is also capable of handling the communication messages

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

between the master processor and the device. The encoders are connected to the cascade, similarly as the decoders. We also need to deliberate that the overflow can occur in several timers at the same time. Thus, the circuit needs to assign priorities to the timers and also has to ensure that the address bus would be written only by one device at the same time. For fulfilling the first requirement we decided to use encoders with priority. The exclusive bus access could be however achieved only by providing feedback from the hierarchically higher priority encoders to the subordinate priority encoders. For such function, we have developed a nonstandard architecture which provides both encoding with priority and the feedback decoding function. The circuit also serves the purpose of propagating the overflow flag. The scheme of the proposed architecture for the address encoding process is shown in Fig 3.

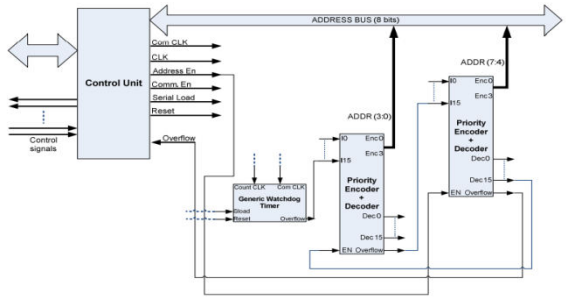

Fig. 3: proposed architecture

IV.RESULTS

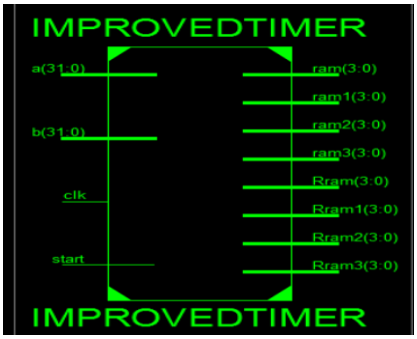

Fig. 4: RTL schematic

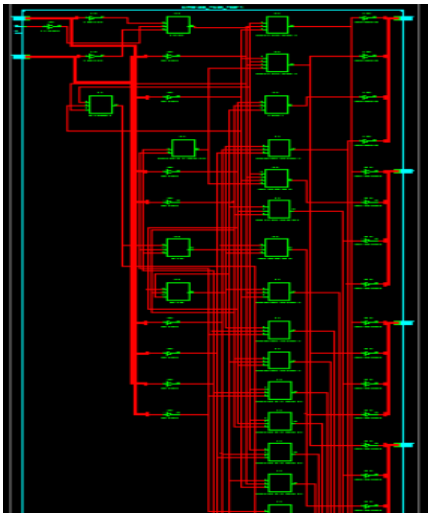

Fig. 5: Technology schematic

Fig. 6: OUTPUT

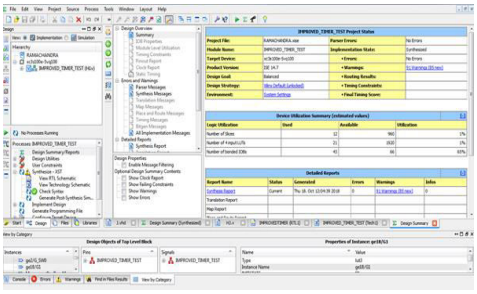

Fig. 7: REPORT

V. CONCLUSION

This paper presented in detail the architecture and design ofan improved windowed watchdog timer and its implementation in FPGA. The watchdog timer runs completely independent of the processor and permits adjusting the timer parameters according to the application. Several fault detection techniques are built into the watchdog for the early detection of erratic software modes. It has the capability to identify the failure type and log it, which can become valuable while debugging. Upon detecting a failure, the

watchdog timer also allows the software sufficient time for saving the debug information, before initiating a reset. Implementing the entire design in FPGA has the advantage of making it adaptable and reusable. HDL based designs are vendor-independent and can be used on different FPGA devices with low overhead. The same design can also be customised for different processors and applications with only minor HDL modifications. In addition, realizing the design in FPGA addresses the component obsolescence issues present in long life cycle embedded systems. The implementation has low complexity and takes up very less amount of hardware resources.

VI. REFERENCES

[1] S. N. Chau, L. Alkalai, A. T. Tai, and J. B. Burt, "Design of a faulttolerantCOTS-based bus architecture," *IEEE Transactions on Reliability*, vol. 48, no. 4, pp. 351–359, Dec. 1999.

[2] V. B. Prasad, "Fault tolerant digital systems," *IEEE Potentials*, vol. 8, no. 1, pp. 17–21, Feb. 1989.

[3] J. Beningo, "A review of watchdog architectures and their application to Cubesats," Apr. 2010.

[4] A. Mahmood and E. J. McCluskey, "Concurrent error detection using watchdog processors - a survey," *IEEE Transactions on Computers*, vol. 37, no. 2, pp. 160–174, Feb. 1988.

[5] B. Straka, "Implementing a microcontroller watchdog with a field programmable gate array (FPGA)," Apr. 2013.

[6] J. Ganssle, "Great watchdogs," *V-1.2, The Ganssle Group, updated January 2004*, 2004.

[7] E. Schlaepfer, "Comparison of internal and external watchdog timersapplication note," *Maxim Integrated Products*, 2008.

[8] P. Garcia, K. Compton, M. Schulte, E. Blem, and W. Fu, "An overview of reconfigurable hardware in embedded systems," *EURASIP Journalon Embedded Systems*, vol. 2006, no. 1, pp. 13–13, Jan. 2006.

[9] G. C. Giaconia, A. Di Stefano, and G. Capponi, "FPGA-based concurrent watchdog for real-time control systems," *Electronics Letters*, vol. 39, no. 10, pp. 769–770, Jun. 2003.

[10] A. M. El-Attar and G. Fahmy, "An improved watchdog timer to enhance imaging system reliability in the presence of soft errors," in *Signal Processing and Information Technology, 2007 IEEE International* Symposium *on*. IEEE, Dec. 2007, pp. 1100–1104.