International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 06 Issue: 03 | Mar 2019 www.irjet.net p-ISSN: 2395-0072

FPGA Implementation of an Improved Watchdog Timer for Safety-Critical Applications

A. Trephena Patricia[1], E. Abinaya[2], S. Harika[3], P. Jasmine Florence Hebciba[4], C. Sumathi[5]

[1]Professor, Dept. of ECE, Panimalar Engineering College, Poonamalle, TamilNadu, India.
[2,3,4,5]UG students, Dept. of ECE, Panimalar Engineering College, Poonamalle, TamilNadu, India.

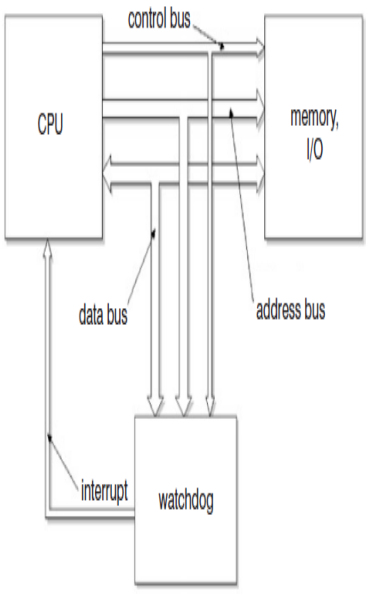
Abstract - *Embedded systems that are employed in safety critical applications require highest reliability. External watchdog timers are used in such systems to automatically handle and recover from operation time related failures. Most of the available external watchdog timers use additional circuitry 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 analyzing the simulation results. Thus after designing the watchdog it is implemented in ATM and verified. The design is validated in a real-time hardware by injecting faults through the software while the processor is executing.*

1. INTRODUCTION

A watchdog timer is an electronic timer that is used to detect and recover from computer malfunctions. Watchdog timers are commonly found in embedded systems and other computer-controlled equipment where humans cannot easily access the equipment or would be unable to react to faults in a timely manner. Watchdog timers may also be used when running un-trusted code in a sandbox, to limit the CPU time available to the code and thus prevent some types of denial-of-service attacks. During ordinary operation, the computer automatically resets the watchdog timer to prevent it from timing out. The timeout signal is used to represent corrective action or actions.

1.1 EXISTING WATCHDOG TIMER:

In the existing system, a watchdog timer with no windowed watchdog is executed. The input is directly sent into the memory, from the memory instructions are processed into the processor, this watchdog will not detect the fault immediately. If there is any error occurrence in between them, it will sequentially wait for its time to trigger the CPU that error has occurred. It is totally dependent on the CPU. Then after CPU, getting the error information it will reset the whole process. It is stated as slow watchdog fault mechanism. The time it takes to reach the error mechanism to rectify is more than the proposed system. Since it is not clock independent, this sequential watchdog is a failure to embedded system. It is rectified during this proposed system.



2. PROPOSED SYSTEM:

An effective watchdog should be able to detect all abnormal software modes and bring the system back to a known state. It should have its own clock and should be capable of providing a hardware reset on timeout to all the peripherals. The watchdog timer proposed in this paper operates independently of the processor and uses a dedicated clock for its functions. A fail flag is raised when the watchdog timer expires and after a fixed amount of time from raising the flag, a reset is triggered. The time in-between can be used by the software to store valuable debugging information to a non-volatile medium.

2.1 WATCHDOG TIMER IMPLEMENTATION IN FPGA:

The design is clocked by its SYSCLK input, which is independent of the processor clock. The possible sets of window lengths are arrived based on the application and hard-coded in the design. These values can be selected by writing to the appropriate bits in the configuration register - SWLEN for the service window and FWLEN for the frame window - after power-on. In order to change the window lengths, the software will have to perform two successive writes to this register with data 0xAAAA and 0x5555. Subsequent to writing the first pattern the second one must be written within 10 µs, after which the software gets a 10 µs period to modify the length configuration fields. If these timings are not strictly met, writes to these bits will remain disabled. The service window is started when a high-to-low transition is detected on the INIT signal. The service window uses a derived clock (SWCLK) that is much slower than the SYSCLK. The slower clock helps in reducing the number of comparators required, thus minimizing the resource utilization in FPGA. The service window has an offset up/down counter that are clocked by the SYSCLK, and a main counter that runs at SWCLK. When the watchdog is correctly serviced, the counters in the service window stop immediately and the frame window starts. The frame window also uses a derived slower clock (FWCLK) for its operations. It has an offset up/down counter and a main counter with functionalities similar to that of the service window. The offset up counter here finds the offset between the termination of the service window and the next rising edge FWCLK. The frame window counters reset when a watchdog service operation occurs within the next service window duration, before the frame window expires

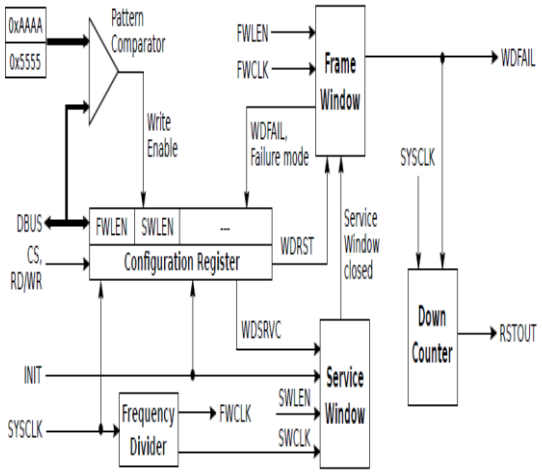
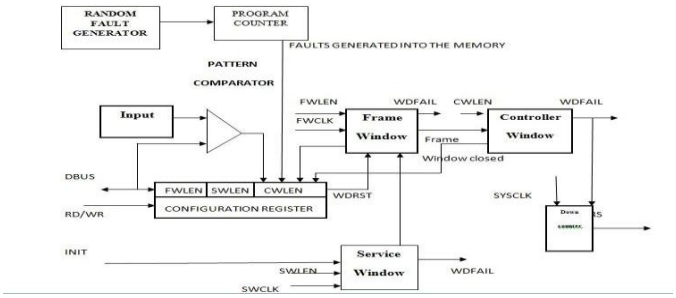


Fig. 6. Functional block diagram of the proposed watchdog timer

2.2 PROPOSED BLOCK DIAGRAM WITH FAULT INJECTION BLOCK

The random numbers generated are injected into the program counter at random periods of time. A random pulse is driven from a second PN sequence generator. This pulse controls a multiplexer whose output is connected to the Program counter. The inputs are either an incrementer or the random generator. At all times the incrementer is selected as this is the normal operation of the system. Simultaneously the watchdog timer is running and a counter records the number of times the watchdog is able to detect the injected faults.



3. IMPLEMENTATION OF WATCHDOG IN SPACE LAUNCH VEHICLE

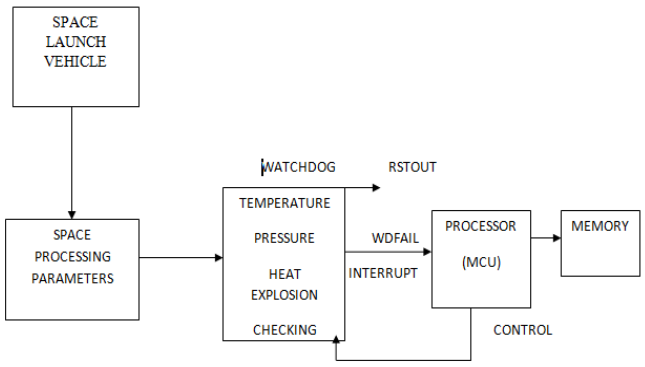
The Space Launch System (SLS) is an American Space Shuttle-derived super heavy-lift expendable launch vehicle. It is part of NASA's deep space exploration plans including a crewed mission to Mars. SLS follows the cancellation of the Constellation program, and is to replace the retired Space Shuttle. The NASA Authorization Act of 2010 envisions the transformation of the Constellation program's Ares I and Ares V vehicle designs into a single launch vehicle usable for both crew and cargo, similar to the Ares IV concept.

Maintaining of space launch vehicle is the most important parameter to be fixed .if there is any parameter not checked or not built; it leads to large loss of money and time. Thus our watchdog timers will do this job in a perfect way.

In our existing system, all the parameters are checked all at a time and if there is any fault in checking of temperature, pressure and heat explosion, then these parameters are made with some terminal value, checked with the parameters pre-recorded, thus leading to watchdog fault if there is an overflow value range.

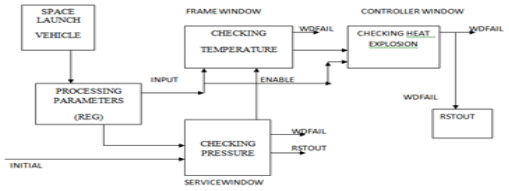
Similarly in our proposed system, we find an efficient way to check our parameters individually, thus leading to windowed watchdog timer. Each window goes on checking with each parameter, thus leading to wdfail in each stage.

3.1 EXISTING SYSTEM



3.2 PROPOSED SYSTEM

Functional block diagram of the PROPOSED watchdog timer



OUTPUT

EXISTING WATCHDOG TIMER

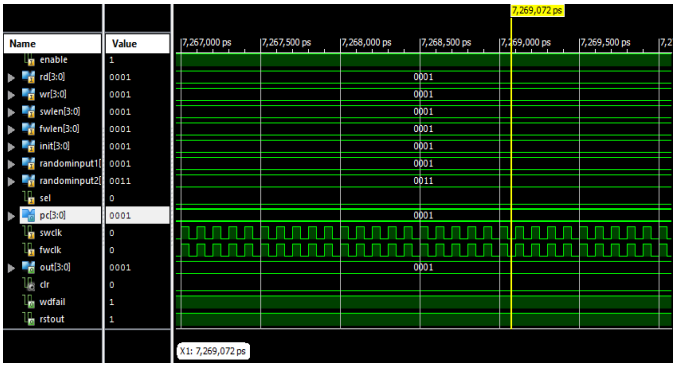
- when wdfail = 1, it will interrupt a failflag to processor, thus processor will enable the watchdog to wdfail state and rstout state.

PROPOSED WATCHDOG TIMER

- when wdfail = 1, it will interrupt a failflag to processor, thus processor will enable the watchdog to wdfail state and rstout state.

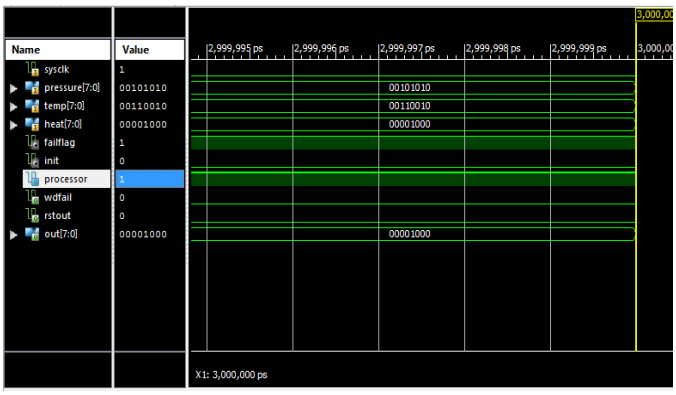
FAULT INJECTION BLOCK WATCHDOG TIMMER

If input and the program counter values updated in register goes to service window then there is a fault. Thus wdfail=1, rstout=1.



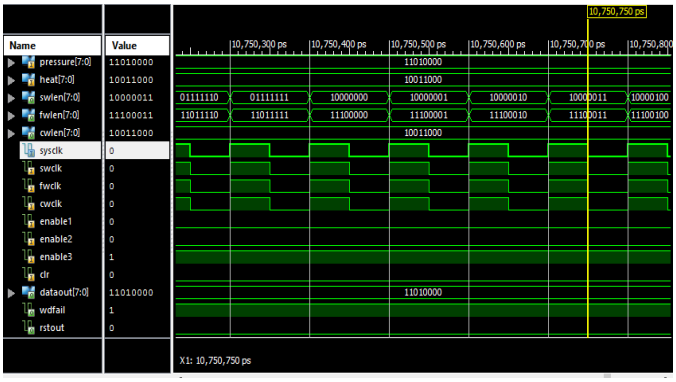
EXISTING SYSTEM APPLICATION IMPLEMENTATION

If parameter of temperature, pressure and heat values does not exceed the limit so there is no error in watchdog. Thus wdfail=0.

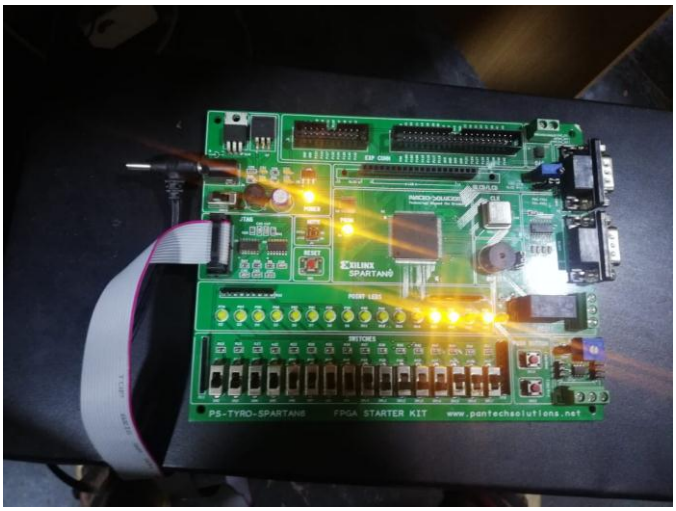


PROPOSED SYSTEM APPLICATION IMPLEMENTATION

If rstout, it goes back to the original initial value of space launch vehicle.



FPGA OUTPUT



CONCLUSION

A good watchdog mechanism requires careful consideration of both software and hardware. It also requires careful consideration of what action to take when the failure is detected. When you design with watchdog hardware, make sure you decide early on exactly how you intend to make best use of it, and you will reap the intend benefits of a more robust system.

ACKNOWLEDGEMENT

We would like to express our special thanks of gratitude to all my teachers as well as our principal who gave us the golden opportunity to do this wonderful project on this topic.

REFERENCES

[1] S. N. Chau, L. Alkalai, A. T. Tai, and J. B. Burt, "Design of a fault tolerant COTS-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 timers application 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.