Journal of Engineering Sciences
Vol 11, Issue 1, JAN/2020
ISSN NO:0377-9254

FPGA IMPLEMENTATION OF AN IMPROVED WATCH DOG TIMER FOR SAFETY-CRITICAL APPLICATIONS

[1]M Keerthi Goud, [2]M Pushpa Latha, [3]Dr. D SubbaRao
[1]PG Scholar, MTech, Dept of ECE, Siddhartha Institute of Engineering & Technology, Ibrahimpatnam, T.S.
keerthimodem@gmail.com
[2]Asso Professor, Dept of ECE, Siddhartha Institute of Engineering & Technology, Ibrahimpatnam, T.S.
[3]Professor & HOD, Dept of ECE, Siddhartha Institute of Engineering & Technology, Ibrahimpatnam, T.S.

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. The design is validated in a real-time hardware by injecting faults through the software while the processor is executing.

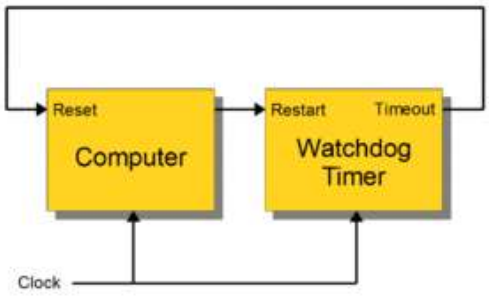
Keywords: FIR, Xilinx, FPGA, Synthesize, Implementation, Simulation.

I. INTRODUCTION

A guard dog clock (here and there called a PC working appropriately or COP clock, or just a guard dog) is an electronic clock that is utilized to recognize and recoup from PC breakdowns. During typical task, the PC normally resets the guard dog clock to keep it from passing, or "timing out". On the off chance that, because of an equipment flaw or program blunder, the PC neglects to reset the guard dog, the clock will slip by and produce a break signal. The break sign is utilized to start restorative activity or activities. The restorative activities commonly incorporate setting the PC framework in a protected state and re-establishing ordinary framework task.

Guard dog clocks are regularly found in implanted frameworks and other PC controlled hardware where people can only with significant effort get to the gear or would be not able respond to deficiencies in an opportune way. In such frameworks, the PC can't rely upon a human to conjure a reboot on the off chance that it hangs; it must act naturally dependent. For instance, remote installed frameworks, for example, space tests are not physically open to human administrators; these could turn out to be for all time impaired on the off chance that they were not able self-sufficiently recuperate from deficiencies. A guard dog clock is generally utilized in cases like these. Guard dog clocks may likewise be utilized when running un-confided in code in a sandbox, to confine the CPU time accessible to the code and in this manner forestall a few sorts of disavowal of-administration assaults.

Constant PC frameworks are characterized as frameworks that are in any conditions ready to ensure their reaction time. Such frameworks are utilized for the most part in different inserted gadgets to ensure their ease of use, for instance to guarantee smooth video playback, and in different modern control applications. Their use in mechanical application is frequently associated with the mission-basic assignments that should be cultivated so as to anticipate framework breakdown or harm. The ongoing PC framework is typically executed on explicit equipment went for such purposes. It can run a basic application that deals with the entire controlled framework or a working framework with a few uses of which everyone has its very own undertaking and reaction due date characterized. One of the techniques to recoup such frameworks from mistake states and guarantee their further usefulness and responsiveness is use of guard dog clocks. Guard dog clock is an equipment gadget normally acknowledged by a counter with match register and explicit framework associations.



The clock can be set to quantify any measure of time inside some sensible limits. This instatement worth is, with a little hold, equivalent to the greatest execution time characterized for the procedure or the entire framework. At the point when the clock is left to flood, it naturally signalizes that a blunder has happened in the program stream or that the program didn't comply with the reaction time constraint. Consequently, when the clock terminates, it makes certain move to recuperate the framework from such mistake state. The activity could be as basic as restarting the framework or as intricate as running a framework demonstrative test. A guard dog clock is said to have terminated on the off chance that it includes not been reset inside a programmable period. It is the job of the specific watched undertaking or procedure to design the guard dog clock and to occasionally reset the clock before it lapses.

The regularly utilized frameworks are typically outfitted with one equipment guard dog clock that is equipped for resetting the entire framework and couple of equipment counters which can furnish the framework with timing data. Continuously working frameworks, when there are various free procedures to be verified, we normally use one of the equipment clocks to give the time base to making virtual guard dog clocks. These virtual clocks are then doled out to every framework procedure that must be observed. Along these lines, every equipment clock is used for the usage of requests of tens to several product guard dog clocks for the simultaneous forms. As the virtual guard dog clocks share one driver for equipment clock, an individual flaw in a procedure or the driver can control the equipment clock in such manner that it is never again usable and along these lines hinders the control for every one of the procedures allocated to this equipment clock.

II. LITERATURE REVIEW

Satellite and Ariel imaging frameworks are situated at high heights. Subsequently, they are more helpless against Soft Errors than comparative frameworks working adrift level. This paper considers the impact of transient blames on microchip-based imaging frameworks. The paper examines the capacity of various guard dog clock frameworks to recuperate the framework from disappointment. Another improved guard dog clock framework configuration is presented. This new plan takes care of the issues of both the standard and windowed guard dog clocks. The guard dog clocks are tried by infusing a flaw while a processor is perusing a picture from RAM and sending it to the VGA RAM for showcase. This technique is executed on FPGA, and outwardly exhibits the presence of quick guard dog resets, which can't be recognized by standard guard dog clocks, and defective resets which happen undetected inside the sheltered window of the windowed guard dog clocks.

We manage verifying the constant frameworks by giving them extra equipment guard dog clocks. This paper proposes the fundamental idea of the numerous equipment guard dog clocks framework and portrays the proposed engineering of the framework giving 256 equipment guard dog clocks. It manages the specific usage of the framework in the FPGA programmable gadget. The outcomes demonstrate that the created framework has a promising potential for improving the security of constant frameworks and that the proposed design is appropriate to be executed in sensibly little programmable gadgets.

A strategy with Observer Pattern and Finite State Machine for guard dog execution is proposed. By utilizing Observer Pattern and Finite State Machine, the guard dog will be advised consequently when the program's state changes. Complex insurance systems can be connected dependent on the change of various states. The reproduction demonstrates incredible enhancement for unwavering quality and viability of the program, particularly for those perplexing projects with perform multiple tasks or multi interfere.

To confront the difficulties coming about because of the expanding thickness of utilization programming parts and higher reliability prerequisites of things to come security frameworks in the car hardware, a trustworthiness programming administration to screen singular application programming segments in runtime is required so as to improve the general framework steadfastness. This paper proposes the use of a Software Watchdog administration giving heartbeat observing and program stream checking. The Software Watchdog is incorporated in a product stage for the car wellbeing hardware. A model-based structure with Mat lab/Simulink and an assessment of this Software Watchdog administration in an equipment on top of it validator are additionally given.

Both standard and windowed guard dog clocks were intended to distinguish stream blames and guarantee the sheltered activity of the frameworks they regulate. This paper examines the impact of transient disappointments on microchips, and uses two techniques to think about the issue inclusion of both guard dog clocks. The principal technique is infusing a deficiency while a processor is perusing a picture from RAM and sending it to the VGA RAM for showcase. This strategy is actualized on FPGA, and outwardly shows the presence of quick guard dog resets which can't be distinguished by standard guard dog clocks, and defective resets which happen undetected inside the protected window of the windowed guard dog clocks. The subsequent strategy is where the deficiency inclusion for every guard dog clock framework is determined. This recreation attempts to mull over numerous components which could influence the result of this examination.

III. METHODOLOGY

EXISTING SYSTEM

FPGA-based simultaneous guard dog for ongoing control frameworks considered the execution of a custom simultaneous guard dog processor in FPGA for continuous control frameworks. The plan did not give a clock to the processor; rather, it played out a sensibility keep an eye on certain factors and a fundamental program stream check. An improved guard dog clock to upgrade imaging framework dependability within the sight of delicate mistake proposed a sequenced guard dog clock that utilized time registers to decide if a deficiency has happened. Be that as it may, it didn't offer much arrangement choices and the deficiency identification highlights actualized were restricted.

PROPOSED SYSTEM:

A successful guard dog ought to have the option to identify all irregular programming modes and take the framework back to a known state. It ought to have its very own clock and ought to be equipped for giving an equipment reset on break to every one of the peripherals. The guard dog clock proposed in this paper works autonomously of the processor and utilizations a committed clock for its capacities. A bomb banner is raised when the guard dog clock lapses and after a fixed measure of time from raising the banner, a reset is activated. The time in the middle of can be utilized by the product to store significant troubleshooting data to a non-unstable medium.

ADVANTAGES:

- A standard guard dog clock can get issues in the framework, for example, draping due to unlimited circles in code execution. In any case, the principle inconvenience of this guard dog is that if the framework enters a flaw state in which it persistently resets the clock, the mistake state will never be distinguished. As such, a standard guard dog clock can distinguish moderate shortcomings, yet can't recognize quick blames which happen inside the guard dog clock period.

- However, a windowed engineering can deal with this appropriately. Here the guard dog characterizes a little time window inside which the guard dog must be reset so as to maintain a strategic distance from a break.

- This gives security against frameworks from running excessively quick and excessively moderate, in this manner expanding the mistake acknowledgment inclusion.

WATCHDOG TIMER ARCHITECTURE

An viable guard dog ought to have the option to identify all strange programming modes and take the framework back to a known state. It ought to have its very own clock and ought to be equipped for giving an equipment reset on break to all the fringe. The guard dog clock proposed in this paper works autonomously of the processor and utilizations a committed clock for its capacities. The design pursues a windowed guard dog usage, where the window time frames can be arranged by the product during introduction. A bomb banner is raised when the guard dog clock lapses and after a fixed measure of time from raising the banner, a reset is activated. The time in the middle of can be utilized by the product to store profitable troubleshooting data to anon-unpredictable medium.

A standard guard dog clock can get issues in the framework, for example, balancing due to unlimited circles in code execution. Notwithstanding, the primary disservice of this guard dog is that if the framework enters a flaw state in which it ceaselessly resets the clock, the blunder state will never be recognized. At the end of the day, a standard guard dog clock can identify moderate deficiencies, however can't distinguish quick blames which happen inside the guard dog clock period. Be that as it may, a windowed design can deal with this appropriately. Here the guard dog characterizes a little time window inside which the guard dog must be reset so as to stay away from a break. This gives assurance against frameworks from

running excessively quick and excessively moderate, accordingly expanding the blunder acknowledgment inclusion.

Watchdog timer input-output interface and configuration register:

Fig. 1 demonstrates the info yield (I/O) interface of the proposed guard dog clock. The guard dog has two yields, to be specific the guard dog bomb yield (WDFAIL) and the reset yield (RSTOUT). At the point when the SYSRESET info is low, the WDFAIL yield stays attested and the RSTOUT yield stays de-asserted.

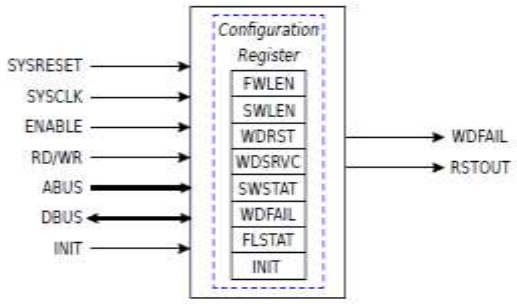


Fig. 1. Watchdog timer input-output interface and configuration register

The control contributions to the guard dog clock, ENABLE and RD/WR, grant the read and keep in touch with the arrangement register. The ABUS and DBUS flag in the figure show address transport and information transport, separately. The length of the two windows can be modified by the product after catalyst by keeping in touch with the bit fields, SWLEN and FWLEN, in the setup register.

When the window time frames are arranged after catalyst, changing the qualities is handicapped by structure. If necessary, the product should experience a stringent open strategy so as to have the option to by and by keep in touch with the setup register. This counteracts any inadvertent alteration of the guard dog window parameters by a runaway code. The INIT contribution to the guard dog clock instates the administration window. A high-to-low change on this information will begin the administration window, gave the bomb banner (WDFAIL). It will promptly close the window and begin the casing window. The edge window characterizes how occasionally the guard dog ought to be adjusted. Regularly, the length of this window is kept marginally more than the primary circle of the installed control framework and the guard dog is overhauled once in each cycle.

Watchdog Timer Initialization

On catalyst or reset the guard dog awakens in a bombed state, i.e., the WDFAIL yield will be declared high. It is the obligation of the product to instate the guard dog and keep it running. Fig. 2 delineates the waveform for guard dog reset introduction and general task. So as to carry the guard dog to a working state, first the guard dog reset (WDRST) field in the design register must be flipped from low-to-high. This, trailed by adjusting the guard dog inside the administration window, will de-affirm the WDFAIL banner and make it operational. Since the casing window is kept bigger than the framework casing time, another administration window will begin before the present casing window terminates. At the point when the guard dog is again appropriately adjusted, the casing window will be reinitialized. For whatever length of time that the casing window counters continue running, no disappointments will be hailed by the guard dog.

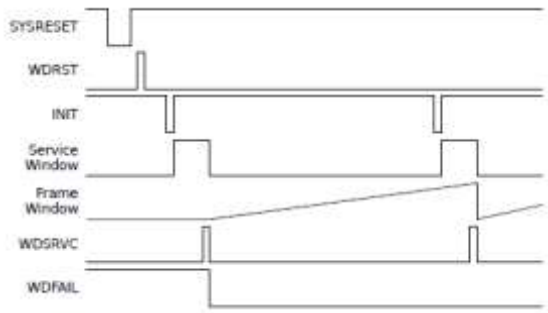


Fig. Watchdog timer initialization and service operation

FAULT DETECTION FEATURES:

A few shortcoming discovery instruments are incorporated with the proposed guard dog clock so as to improve its adequacy in catching unpredictable programming modes. At the point when the product neglects to support the guard dog inside the administration window, the window terminates and sets a bomb banner inside. For this situation, the edge window does not reinitialize and terminates after achieving its terminal worth. On the expiry of the edge window the guard dog attests its WDFAIL signal, demonstrating a disappointment. This disappointment mode is delineated in Fig. 3.

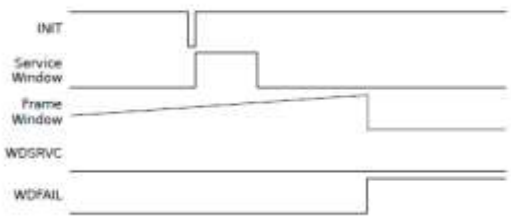


Fig. Watchdog fails due to frame window expiry

Journal of Engineering Sciences
Vol 11, Issue 1, JAN/2020
ISSN NO:0377-9254

A guard dog bomb will happen when the product benefits the guard dog outside the administration window, as appeared in Fig. 4. It tends to be seen that the invalid administration activity in a split second ends the edge window and declares the WDFAIL signal. A good result of this component is that two progressive administration tasks will likewise prompt a guard dog fall flat. Here, the main administration activity will quickly close the administration window and the following one will perpetually happen outside the window. This ends up identical to adjusting the guard dog outside the administration window and prompts a guard dog disappointment.

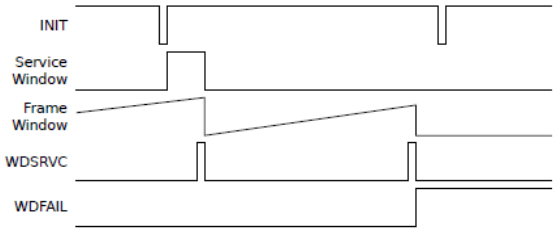


Fig. Watchdog fails due to service outside the service window

Fig. 5 delineates a situation where the WDSRVC falling edge is happening inside the administration window. This is likewise considered as an unlawful administration task and the guard dog bomb sign is stated. This infers, in the wake of overhauling the guard dog, the product is required to de-affirm the WDSRVC signal before the beginning of the following administration window. These flaw discovery instruments guarantee that a product running haywire won't go undetected by the proposed guard dog clock.

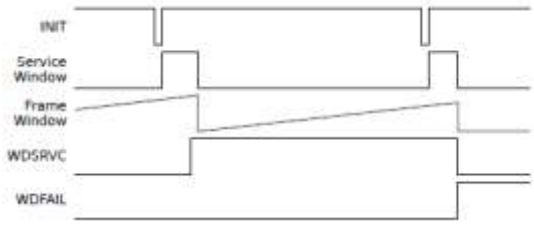
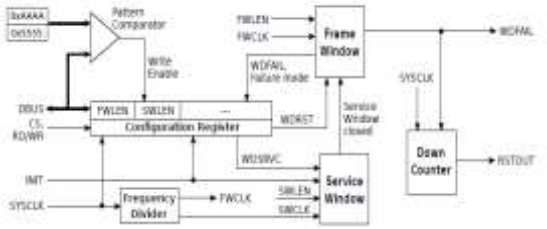


Fig. Watchdog fails due to WDSRVC falling edge inside service window

WATCHDOG TIMER IMPLEMENTATION IN FPGA:



The structure is timed by its SYSCLK input, which is free of the processor clock. The potential arrangements of window lengths are arrived dependent on the application and hard-coded in the structure. These qualities can be chosen by keeping in touch with the fitting bits in the design register - SWLEN for the administration window and FWLEN for the edge window - after power-on. So as to change the window lengths, the product should perform two progressive keeps in touch with this register with information 0xAAAA and 0x5555. Ensuing to composing the principal design the subsequent one must be composed inside 10 µs, after which the product gets a 10 µs period to change the length arrangement fields. On the off chance that these timings are not carefully met, keeps in touch with these bits will stay impaired.

The administration window is begun when a high-to-low change is recognized on the INIT signal. The administration window utilizes an inferred clock (SWCLK) that is much slower than the SYSCLK. The slower check helps in decreasing the quantity of comparators required, along these lines limiting the asset usage in FPGA. The administration window has a balanced up/down counter that are timed by the SYSCLK, and a fundamental counter that keeps running at SWCLK. At the point when the guard dog is effectively adjusted, the counters in the administration window stop promptly and the casing window begins. The casing window additionally utilizes an inferred slower clock (FWCLK) for its tasks. It has a counterbalanced up/down counter and a fundamental counter with functionalities like that of the administration window. The balance up counter here finds the counterbalance between the end of the administration window and the following rising edge of the FWCLK. The casing window counters reset when a guard dog administration task happens inside the following administration window span, before the casing window terminates.

Reset Initialization and Fault Detection

Journal of Engineering Sciences
Vol 11, Issue 1, JAN/2020
ISSN NO:0377-9254

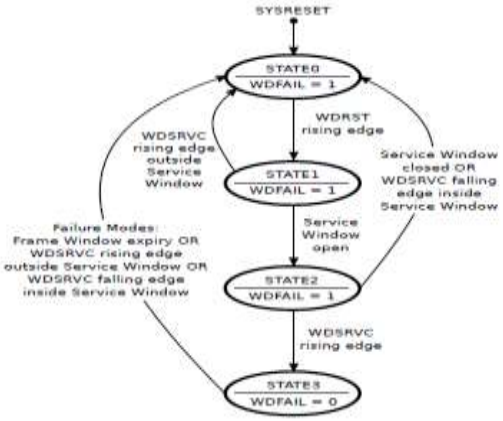


Fig. Finite state machine design of watchdog fault detection logic

On catalyst the WDFAIL yield is attested, demonstrating a guard dog disappointment. A rising edge on the WDRST bit readies the guard dog clock for instatement. At the point when the administration window opens, a rising edge on the WDSRVC bit deasserts the WDFAIL yield and the window counters begin running. Be that as it may, if the guard dog is overhauled inaccurately, the entire introduction procedure is disposed of and the product should rehash the whole strategy.

The WDFAIL sign gets de-affirmed just when the guard dog is appropriately instated. Statement of the guard dog bomb likewise triggers a reset counter that keeps running for a predefined measure of time. The span of the counter can be controlled by considering the measure of troubleshoot data that should be put away. On the expiry of the counter, the WDT states its RSTOUT yield high. The reset counter will be nonfunctional during catalyst and the RSTOUT yield will be set to low now. At the point when the guard dog is instated just because, the counter gets naturally empowered.

IV. RESULTS AND DISCUSSION

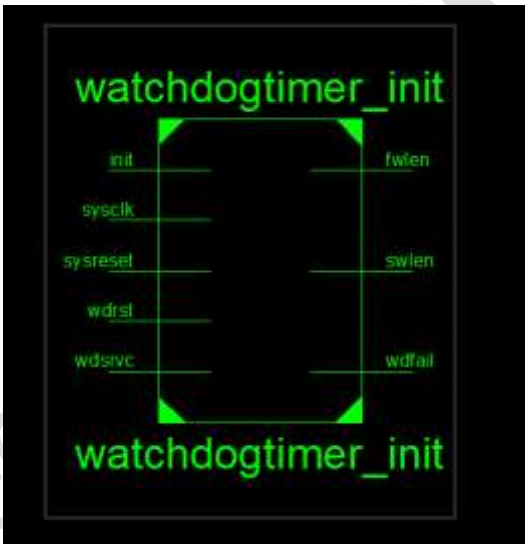


Fig. RTL schematic of dff module

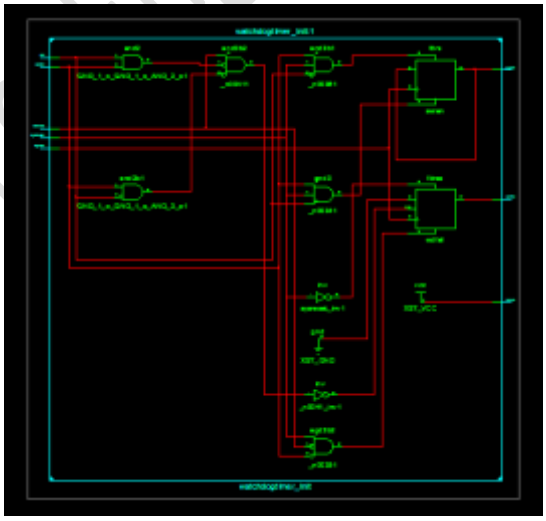
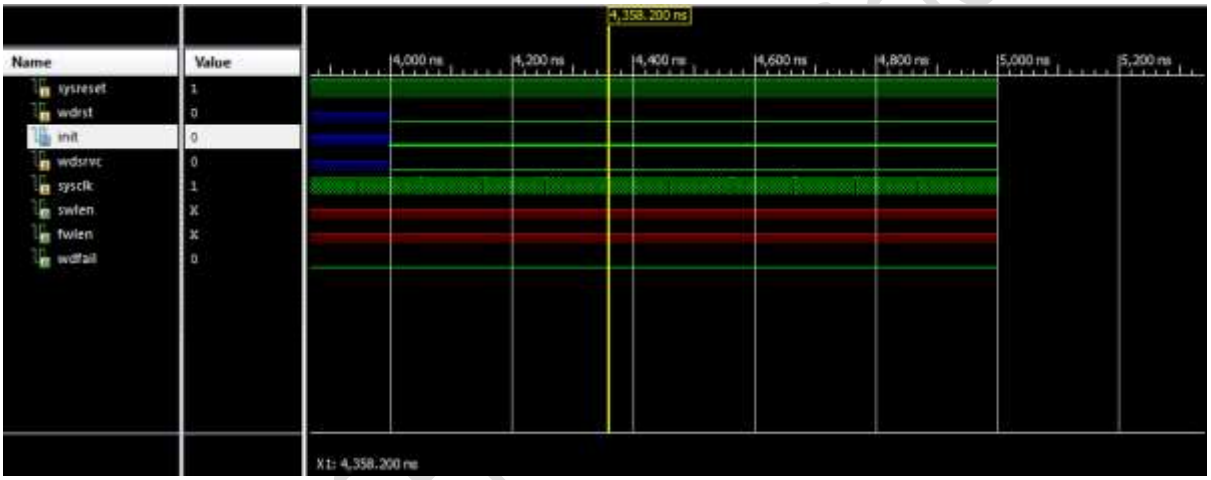


Fig. RTL schematic

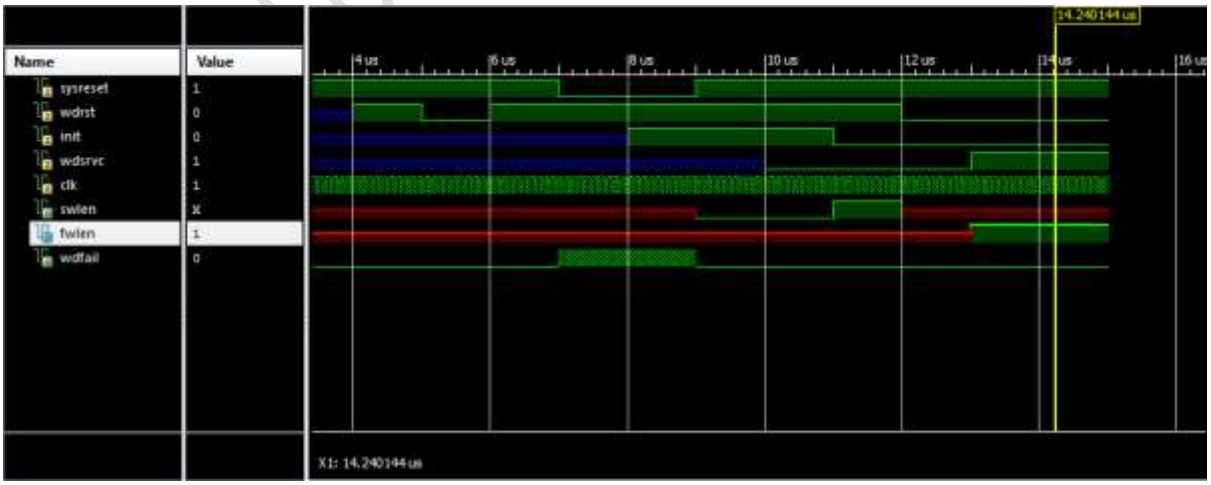
Sytreset=0, wdfail= 1;

Journal of Engineering Sciences
Vol 11, Issue 1, JAN/2020
ISSN NO:0377-9254

To start the watchdog, wdrst= 0 to 1;
wdsrvc= 0, to initialize the service window.



To start the service window, wd=0, then swlen=1, thus fwlen= 1'bx



Overall output:

Design summary:

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	2	11440	0%
Number of Slice LUTs	4	5720	0%
Number of fully used LUT-FF pairs	0	6	0%
Number of bonded IOBs	8	102	7%
Number of BUFG/BUFGCTRLs	1	16	6%

V. CONCLUSION

This paper exhibited in detail the engineering and plan of an improved windowed guard dog clock and its usage in FPGA. The guard dog clock runs totally free of the processor and licenses altering the clock parameters as indicated by the application. A few flaw location methods are incorporated with the guard dog for the early discovery of whimsical programming modes. It has the capacity to recognize the disappointment type and log it, which can end up significant while troubleshooting. After distinguishing a disappointment, the guard dog clock additionally permits the product adequate time for sparing the investigate data, before starting a reset.

VI. 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 timer's 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 Journal on 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.

[11] M. Pohronsk´a and T. Kraj˘covi˘c, "FPGA implementation of multiple hardware watchdog timers for enhancing real-time systems security," in *EUROCON-International Conference on Computer as a Tool (EUROCON), 2011 IEEE*. IEEE, Apr. 2011, pp. 1–4.

[12] H. Guzman-Miranda, L. Sterpone, M. Violante, M. A. Aguirre, and M. Gutierrez-Rizo, "Coping with the obsolescence of safety- or mission critical embedded systems using FPGA's," *IEEE Transactions on Industrial Electronics*, vol. 58, no. 3, pp. 814–821, 2011.

[13] H. Amer and A. Sobeih, "Increasing the reliability of the Motorola MC68HC11 in the presence of temporary failures," in *Electrotechnical Conference, 2002. MELECON 2002. 11th Mediterranean*. IEEE, May 2002, pp. 231–234.

[14] A. M. El-Attar and G. Fahmy, "A study of fault coverage of standard and windowed watchdog timers," in *Signal Processing and Communications, 2007. ICSPC 2007. IEEE International Conference on*. IEEE, Nov. 2007, pp. 325–328.

[15] M. Barr, "Introduction to watchdog timers," *Embedded Systems Design*, 2001.

[16] N. Murphy, "Watchdogs timers," *Embedded Systems Programming*, p. 112, 2000.

[17] F. Afonso, C. A. Silva, A. Tavares, and S. Montenegro, "Application level fault tolerance in real-time embedded systems," in *Industrial Embedded Systems, 2008. SIES 2008. International Symposium on*. IEEE, Jun. 2008, pp. 126–133.

[18] M. Wirthlin, "High-reliability fpga-based systems: Space, high-energy physics, and beyond," *Proceedings of the IEEE*, vol. 103, no. 3, pp. 379–389, Mar. 2015.

[19] H. Ziade, R. A. Ayoubi, R. Velazco *et al.*, "A survey on fault injection techniques," *The International Arab Journal of Information Technology*, vol. 1, no. 2, pp. 171–186, Jul. 2004.

AUTHOR PROFILE'S

MODEM KEERTHI GOUD, is a proficient PG Scholar, Master of Technology, Dept of ECE, VLSI & ES, Siddhartha Institute of Engineering & Technology (SIET), JNTUH, Ibrahimpatnam, T.S. Along with initial degrees of Bachelor of Technology in Electronics and Communication Engineering (ECE) from Brilliant Institute of Engineering and Technology (BRIL), JNTUH, Abdullapurmet, Hyderabad. She is Currently bounded and working as a Testing Engineer in Research Centre Imarat (RCI), Defence Research and Development Organization (DRDO), Kanchanbagh, Hyderabad.

M PUSHPALATHA working as associate professor in ECE branch, Siddhartha Institute of Engineering and Technology, Hyderabad, TS, India. She has completed her Master's degree with specialization in Digital Electronics and communication systems from Mahaveer Institute of Science & Technology affiliated to JNTU Hyderabad in the year 2008. Prior to this has completed bachelor's degree in Electronics and communication engineering from G. Narayanamma Institute of Technology and Science affiliated to JNTU Hyderabad. Pursuing Ph.d in the area of wireless communication, Sri Satya Sai University of technology and medical Science, Bhopal, Madhya Pradesh Her carrier started as a lecturer and has total 7 years of experience in teaching field. Out of which 3 years worked as Assistant Professor and 4 years as Senior Assistant Professor in a single organization named Abhinav Hi-Tech College of Engineering. Presently working as an Associate Professor for Siddhartha Institute of Engineering and Technology in Department of Electronics and Communication Engineering. She attended and also conducted many workshops and conferences. She is very much interest to do research on DECS, wireless technology and signals.

DR. D SUBBA RAO, is a proficient Ph.D person in the research area of wireless communications from Rayalseema University, Kurnool along with initial degrees of Bachelor of Technology in Electronics and Communication Engineering (ECE) from Dr. SGIET, Markapur and Master of Technology in Embedded Systems from SRM University, Chennai. He has 16 years of teaching experience and has published 98 Papers in International Journals, 2 Papers in National Journals and has been noted under 4 International Conferences. He has a fellowship of The Institution of Electronics and Telecommunication Engineers (IETE) along with a Life time membership of Indian Society for Technical Education (ISTE). He is currently bounded as an Associate Professor and is being chaired as Head of the Department for Electronics and Communication Engineering discipline at Siddhartha Institute of Engineering and Technology, Ibrahimpatnam, Hyderabad.