International Journal of Engineering and Advanced Technology (IJEAT)
ISSN: 2249-8958 (Online), Volume-8 Issue-5S3, July 2019

Predicting Reliability of Open-Source Hardware Platform

Nikhil S Damle, Gauri Damle, Rushikesh Deshmukh

Abstract*: Embedded system today are influencing every sphere of life and all application domains. These systems were initially designed according to the application requirement as special hardware. This approach of embedded system design was influenced by particular family of processor. The new approach of embedded system design envisages the use of general purpose on the shelf hardware modules to be incorporated. These on the shelf hardware modules are also referred open source hardware platforms as their entire hardware module design in available in public domain. Thus populating such hardware platforms by different individuals becomes easy. At instances these open source modules are not as good as the original ones. This is because calculating and predicting the successes achieved or failure caused by the copy is not tested. This can be verified by the performance of the module designed in terms of reliability.

***Keywords*— Open source, firmware Software Reliability; Reliability Testing; module;**

I. INTRODUCTION

Open-source hardware are prominently becoming popular as number of companies providing a basket of open-source electronics which can also be used for electronics prototyping. These prototyping platforms can be utilized for control as well as other embedded applications. All these variants of open source electronics as only as reliable as their building blocks. In this paper an open source hardware prototyping platform is considered for an open source application development problem.

II. HARDWARE DESCRIPTION: ARDUINO

The paper discusses the use of Ardunio-UNO a popular open source hardware platform for motion control. Ardunio-UNO is an Atmega328p based development board which can be connected to the host system via a USB cable for application development. The application development is monitored through the Arduino Software (IDE). The motion control profiles are decided according to the block diagram. The motion profiles are used for to controlling operations like drilling, milling and engraving.

A. Basic test setup

The open source hardware application development system consists of the processor board, peripherals power supply and interconnects. To ensure the system performing seamlessly the system performance needs to test against some standard sets.[1]

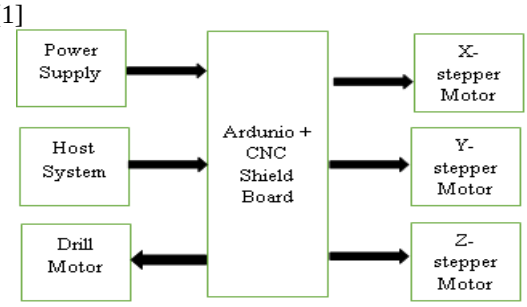

Figure 1. Open source Hardware Module interconnection

Arduino board is connected to the CNC shield, with stepper motor driver A4988. The motion control profiles are fed to the CNC shield. A power source of 24V is used for the setup. The stepper motor used are required to provide sufficient amount of torque to complete the motion profiles. Thus the initial current thrust will be generated with some amount of heat dissipated through the heat sinks. The stepper motor takes one step of 1.8 degree. It requires 200 step for a complete revolution. Due to X axis stepper motor, the wooden board moves back and forth. Due to Y axis stepper motor, the spindle moves left and right. The depth to the spindle is provided by the Z axis stepper motor. The spindle requires a DC motor for the drilling purpose. [2],[3]

B. Reliability Model of Open Source platform

The reliability of *Open Source platform* depends on the reliability of the components individually as well as a whole. Every electronics circuit in use is exposed to all types of external stress due to environment is used in i.e. physical stress on electronic device being dropped, the thermal stress of temperature differences and the electrical stress applied when the device is powered up.[4]

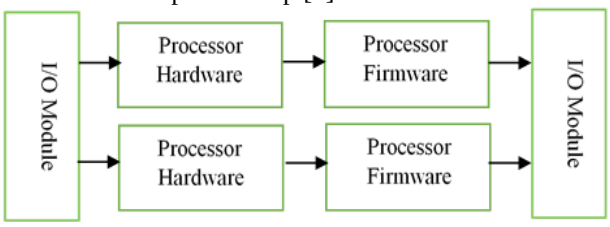

Figure 2 Hardware redundant Reliability Model

Reliability tests are designed for selection of parameters that are in accordance working environment, for accurate assessment of product reliability. The reliability model of system is considered with parallel and serial connectivity.

Manuscript published on 30 July 2019.
* Correspondence Author (s)
Nikhil S Damle, Assistant Professor Department of Electronics Engineering, RCOEM, Nagpur India.
Gauri Damle, Research Engineer, RIVA LABS CVIN-VNIT Nagpur, India
Rushikesh A Deshmukh, Assistant Professor Department of Electronics Engineering, RCOEM, Nagpur India

Following important considerations are required to be noted here:

- The processor form the open source board is considers or modelled as hardware processor and firmware processor.
- The firmware and hardware are running in a sequential manner because the processor cannot complete its function unless the hardware or the part of firmware is not operational.
- The firmware and the hardware are joint unit. In this unit the spares are placed in concurrent manner to provide a parallel operation of the system even is case of failure of processor.
- The I/O's, the processing unit are sequentially running leading to absolute failure of the system.[5]

C. Hardware failures

Operations of the system can cease to occur due the failures.

1. Failures due to inherent flaws in the designs.
2. Infant Mortality is a caused due to newly manufactured hardware failure
3. Random Failures can be seen throughout the operation cycle of hardware. Leading to system malfunction.
4. Redundancy is provided to address the issues of these types of failures.
5. Wear Out: When a hardware module has aged the degradation of component may cause the hardware to fail.

The faults of this nature can be eliminated by preventive maintenance.

For open source systems it is important to analyze the software component failures. Failures in the software or firmware are removed by keeping track of software defect in the system and upgrading the firmware from time to time in event of hardware changes. [5]

D. Process of Software design & defect detection

The process used to develop the firmware mainly consists of initial design and code.

1. Intricacy of the software
2. Extent of the software
3. Understanding of the team developing the software
4. Proportion of code reused from a previous stable project
5. Thoroughness of testing before shipment.

Defect density is measured in number of defects per thousand lines of code (defects). [6]

III. OPEN SOURCE RELIABLE OPERATIONS

A system cannot be viewed as separate hardware and software. But a combination of both running together with failures and successes taken together. When the amount of successes is more the system can provide reliable operation. [7]

A. Reliability Parameters

The performance of embedded system is determined on the basis of the successes that the system attains. The attainment of the successes rate of any system is measured in terms of the time the system needs to recover from a failure to attain success. In other words, Mean Time between Failures (MTBF) is parameter used for determination of the time between consequent failures of the hardware. MTBF for software is also essential component of the system reliability. It can also be expresses as a multiple of defect rate with KLOCs executed per second. In the world of embedded system the system performance is important at the same time the system downtime should be also minimum. The system downtime is avoided by providing redundancy. But unfortunately all embedded system is not designed with the feature of redundancy. The system can be designed with replaceable or repairable hardware software modules. [8], [9]

The hardware will be further divided into functions like power supply, bus architecture, interfaces like inputs and outputs and control. All these functions except the power supply is a purely a hardware operation while all other functions are require an application program for smooth operations of the desired function as in table 1(a).

Table 1 a Partitioning of functions

Function	Hardware	Software
Power Supply	Hardware	-
Bus Architecture (Arduino Uno Board)	Hardware	Software
Interface input-output (Sensor-Actuators)	Hardware	Software
Control (Arduino board)	Hardware	Software

The amount of time required to repair the failed component of an operational system literally means replacing the component. The replacement/repair work can be estimated based on the availability of the component. [10]

Table 1(b), (c) are evident of the time required for the replacement based on the availability of the redundant component on the shelf. It is also important to understand and estimate the need for replacement. The MTTR of the hardware as well as the software module or modules has to be estimated based of the need. MTTR for a piece of software can be calculated considering the time required for rebooting once a software fault is detected.

$$MTBF = \frac{T}{R} \quad (1)$$

where T = total time and R = number of failures

Table 1(b) Estimating the Hardware MTTR

Estimating the MTTR for Open Source Hardware		
Placement of Hardware Spares	**Floor Management**	**MTTR**
Shop Floor	24 hours a day	30 Minutes
On-site	Regular working	3 days
Off- site	Operator informed in event of fault	7 days

If the MTBF values are known then the MTTR can be found out, the availability of the component can be determined.

Table 1(c) Estimating the Software MTTR

Software MTTR Estimation		
Software fault recovery	**Software reboot on Fault detection**	**Estimated MTTR**
Software failure is detected by watchdog	Processor reboots form ROM	30 seconds
Watchdog detecting software failure	Processor reboots, firmware download from host system	10 minutes
No detection	Manually reboot	30 minutes to 2 weeks

B. Availability & Downtime of system

Availability of the open source module and downtime of the system is important in deciding the systems operational time. Availability is a term can be expressed as the percentage of time when system is running smoothly. Availability of a hardware/software is determined obtained by the formula given below.

$$Availability = \frac{MTBF}{MTBF + MTTR} \quad (2)$$

$$Reliability = e^{-t/MTBF} \quad (3)$$

The estimated values of the availability are computed for comparison with the relation of failure time and time to repair. It is evident from the graph that the availability of the software is higher as compared to hardware therefore contributing less downtime for the software defects.

Table 2(a) Reliability prediction using MIL-217 @ T=25^0 C

Component	Specs	MTBF	Failures/ 106 Hrs	FIT
Xmer	MIL	173913043.14	0.00580	5.75
Diode	Switching	132978723.65	0.00750	7.52
Diode	Voltage Reg	37878788.08	0.02640	26.4
Diode	Fast Recovery	1923076.99	0.52000	520
Up		28904162.00	0.34500	34.5061
EEPROM	256k	374330387.00	0.00270	2.64714
Resistor	RC	66666660.87	0.00150	1.5
Resistor	Lower	199999985.84	0.00500	5
Transistor	<1W Plastic	95238089.39	0.00010	0.105
Transistor	>1W Plastic	250626559.27	0.00400	3.99
Solder	100	3846153.99	0.26000	260
Solder	200	1923076.99	0.52000	520
Rotating Motor	DC	624999.99	1.60000	1600
Crystal		14880592.91	0.06720	67.2
DC Lamp		76923.00	13.0000	13000
Relay	Armature	854700.00	1.17000	1170
Capacitor		199999985.84	0.00500	5.4
Connector	PCB-One Piece	185185180.38	0.00190	1.9
Connector	IC Socket	526315780.60	0.00190	1.9
LED		3039513657	0.00030	0.239
SSD		230414747.34	0.52000	520
78xx		81672083.00	0.01220	12.2
Switch	PTT	49999996.46	0.02000	20

Table 2(a) Reliability prediction using MIL-217 @ T=50^0 C

Component	Specs	MTBF	Failures/ 106 Hrs	FIT
Xmer	MIL	1142857108	0.0009	0.875
Diode	Switching	132978723.7	0.0075	7.52
Diode	Voltage Reg	37878788.08	0.0264	26.4
Diode	Fast Recovery	1923076.99	0.52	520
Up		7018694	0.1425	142.4766
EEPROM	256k	37433087.65	0.0027	2.6714
Resistor	RC	666666660.9	0.0015	1.5
Resistor	Lower	199999985.5	0.005	5
Transistor	<1W Plastic	833333293.8	0.0012	1.2
Transistor	>1W Plastic	21929824.19	0.0456	45.6
Solder	100	2758310.15	0.3625	362.5408
Solder	200	2030288.63	0.4925	492.5408
Rotating Motor	DC	624999.99	1.6	1600
Crystal		14880952.91	0.0672	67.2
DC Lamp		76923.08	13	13000
Relay	Armature	854700.00	1.17000	1170
Capacitor		199999985.8 4	0.00500	5
Connector	PCB-One Piece	92592590.19	0.0108	10.8
Connector	IC Socket	263157890.3	0.0038	3.8
LED		265957447.4	0.0038	3.76
SSD		20161289.74	0.0496	49.6
78xx		53903573.34	0.0186	18.5516
Switch	PTT	49999996.46	0.02000	20

Table 2 (a) and 2 b) indicates that the mean time between failure (MTBF) and the failure rate of the hardware modules is a major link in calculating the operating availability of the hardware component at a given operating temperature. Here the operating temperature is 25^0 C. The reliability of the hardware component can be then calculated in terms of the working hours or the reliability of the hardware module.

C. Processor Firmware Reliability

The firmware design depends on the software architecture preferred by the designer. Each of the software architectures has their own advantages in terms of simplicity and fast operation or reliable and slow operation. To identify the best suitable software architecture for the embedded system depends on the software's reliability which is entirely a separate study. The I/O's are easy to manage therefore they can be made available without putting them in a redundant array.

IV. SYSTEM AVAILABILITY CALCULATION

Once the availability and downtime of the component is calculated based on how the use of the component is easy to predict. Estimating the availability of the system as a uniform entity. The system availability and downtime analysis depends on the consideration that under what condition the components of the system are supposed to interact with each other.

The hardware components are running sequentially while the software is considered to operate in parallel with the hardware modules. [11].

Once the reliability of the individual hardware component is calculated it is not very difficult to predict the reliability of an embedded system configuration. The configurations of embedded systems were considered with different hardware components.

Figure 4 (a),(b),(c),(d) shows the cumulative effect of MTBF of the hardware components on the reliability of the embedded system.

V. RESULTS

A. Reliability Test under MIL-217F-1

MIL-217 is widely used and accepted method for predication data for electronic components leading to "Reliability Prediction of Electronic Equipment. The determination of reliability of a system is important when is system is to be operated for a longer time. In the early stages of development it is done using the Parts Count method. It does its analysis based on the type and the number of the components used in the design. [12]

Table 3 MIL-217 Part count Method Reliability

Function	MTBF	Availability	Reliability
Power Supply	772251755.165	0.999999782	0.9999999987
Processor I/O	26300598415.15	0.999999994	1.0000000000
Processor Display	7642145200.69	0.999999978	0.9999999999
Processor Bus Interface	1840366847.83	0.999999909	0.9999999995

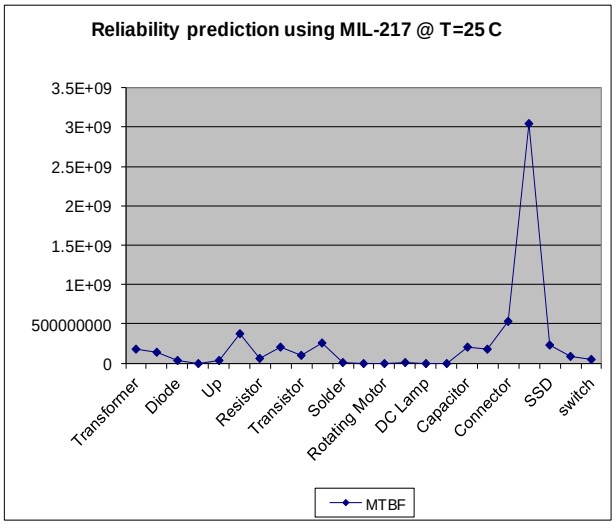

Figure 4(a) MTBF Calculation using MIL-217 @ T=25 C

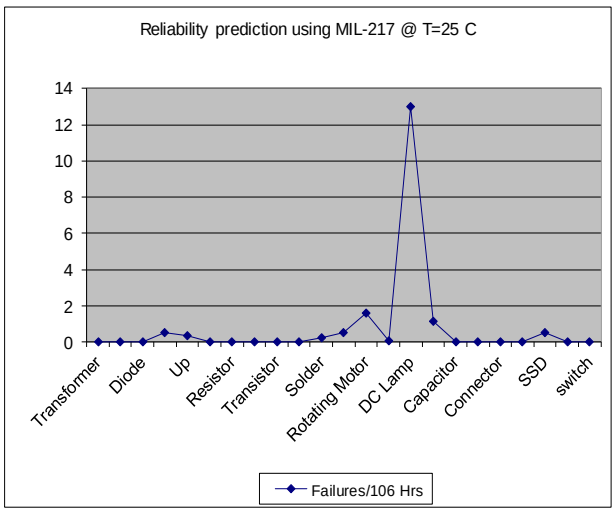

Figure 4(b) Failure rate Calculation using MIL-217 @ T=25 C

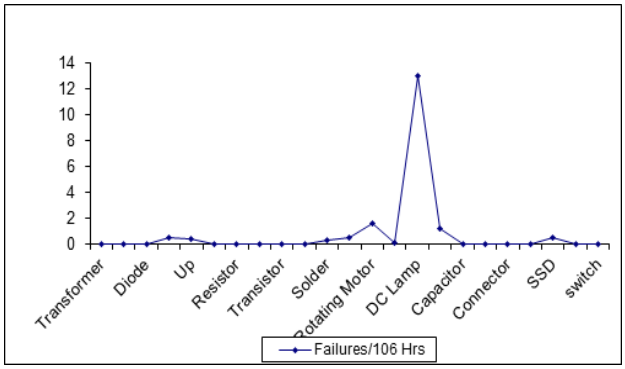

Figure 4(c) Reliability prediction using MIL-217 @ T=50 C

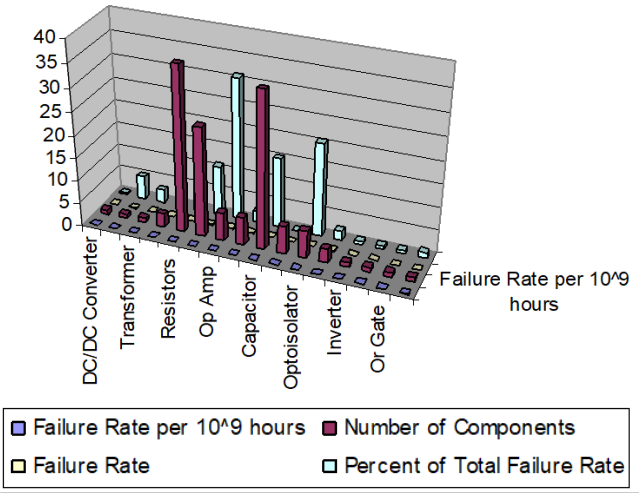

Figure 4(d) Reliability prediction using MIL-217 @ T=50 C

VI. CONCLUSION

The system after reliability check and verification is employed. Efficient use of open source on the shelf modules are simplifying the actual system development. The system is reliable, offers repeatability due to low downtime of critical components used.

The availability of these spares or the redundancy offers better control over the system configuration of sensors-actuators; accuracy of motion controllers & safety devices. In the initial product development phase it provides a quick estimate of hardware components required. The analysis of system based on the component setup.

The prediction also considers the quality factor and learning factor i.e. manufacturing standard and number of years the component has been in production. The prediction results are more accurate than a simulation proving more useful with the design is complete.

FUTURE SCOPE

This fact can be used in determining the relationship between the software reliability of the open source hardware and actual MTBF calculated for such firmware executed on the platform with similar reliability prediction setup.

The results of these test can be a useful is comparing it with the benchmarking firmware available for open source hardware platform where only the execution time, power consumption can be the primary factor of test.

Comparative study of the open source boards based on the available onboard resources against each other in benchmarking test to check their actual performance will be carried out. Until now the power consumption was treated as a totally hardware dependent issue, but now-a-days embedded system software takes care of power management.

Power consumption (mAH) is measured using a USB Power Meter over 60 minutes while running the benchmarking blink sketch at the default 16 MHz clock speed on different open source boards. The NANO board consumed 10 mAH power, where UNO consumed almost the same 12 mAH power. On the other hand, Mega took almost double the power consumed by NANO and UNO. The detailed comparison on factors like computation power, boot up time and power consumption will support the open source hardware platform claim in higher reliability.

REFERENCES

1. P. J. Gil, G. Benete, J.J. Serrano "Design of an Intelligent Stepper Motor Controller Module into a Distributed Process Control Architecture" MELECON '94. Mediterranean Electro technical Conference 12-14 April 1994
2. Erich Sulzer, Michael Bertsch, "Design & Engineering of Modern Process Control System" 1997 IEEE/PCA Cement Industry Technical Conference. XXXIX Conference Record (Cat. No.97CH36076), 20-24 April 1997
3. Fabiano Hessel, Vitor M. da Rosa, Igor M. Reis, "Abstract RTOS Modeling for Embedded Systems" 15th IEEE International Workshop on Rapid System Prototyping (RSP'04) PP 210-216
4. Semiconductor Device Reliability Failure Models Technology Transfer # 00053955A-XFR International SEMATECH *May 31,2000*
5. Dr. Peter Tröger, Krishna B. Misra: Reliability Prediction -Handbook of Performability Engineering. Springer. 2008 (p. 265ff)
6. Andreas Gerstlauer Haobo Yu Daniel D. Gajski "RTOS Modeling for System Level Design" Design, Automation and Test in Europe Conference and Exhibition (DATE'03)
7. Zhengting He "Timed RTOS Modeling for Embedded System Design" 11th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS'05)
8. Cyprian F. Ngolah, Yingxu Wang, and Xinming Tan "Implementing Task Scheduling and Event Handling In RTOS+"CCECE 2004- CCGEI 2004, Niagara Falls, May 2004
9. Nikhil Shirish Damle, Avinash G Keskar "Co-simulation of Networked Embedded System: Verification Approach" SCIS & ISIS SCIS & ISIS 2008, pp 2086-2090
10. Chen, Hui & Qin, Zhidong. (2010). Reliability demonstration testing method for embedded operating systems. 2nd International Conference on Software Engineering and Data Mining, SEDM 2010.
11. Farah Lakhani, Michael J. Pont Farah Lakhani, Michael J. Pont Applying design patterns to improve the reliability of embedded systems through a process of architecture migration, IEEE 14th International Conference on High Performance Computing and Communications 978-0-7695-4749-7/12 $26.00 © 2012 IEEE DOI 10.1109/HPCC.2012.228
12. MIL-HDBK-217 (Electronics Reliability Prediction) Handbook. 2007

ACKNOWLEDGMENT

We wish to acknowledge founders of Arduino for providing such a wonderful and easy to learn platform to electronic hobbyists, artists, designers, makers; basically everyone who wants to make things. We also extend our acknowledgement to the huge Arduino open source community to provide library support and documentation required to write this paper.

AUTHORS PROFILE

Nikhil S Damle completed his B.E (Electronics) from Nagpur University Maharashtra; M. Tech Electronics & Communication Engineering from VNIT Nagpur having specialization in Networked Embedded System; Master's Degree in Business Administration with specialization in Information Technology and Systems and Post Graduate Diploma in Embedded Systems and Cyber Law. He has also worked Research Assistant in VNIT, Nagpur for over two years. He has teaching experience of 15 years. He is recipient of Certificate of Achievement from Deakin University for his achievements in course on Cyber Security for Small & Medium Enterprises: Identifying Threats & Preventing Attacks. He has published 16 papers in journals and conference of national and international repute. He is a MIEEE, Chartered Engineer [India], MIE, MIETE, MCET (I). His areas of interest include Computer Networks, Embedded System design, Instrumentation, Continuing education and skill development.

Gauri N Damle completed her B.E. (Electronics & Telecommunication) from RTM Nagpur University Maharashtra; M. Tech Electronics & Communication Engineering from RTM Nagpur University. She has published few research papers in journals/conferences of repute. She is currently responsible for Design Engineering Division for RIVA Labs Nagpur a technology start-up incubated at Center of VLSI and Nanotechnology VNIT, Nagpur. Currently engaged in design and development of

Rushikesh Deshmukh graduated from College of Engineering Pune and is recipient of Gold Medal in M. Tech VLSI Design. He has worked as Research Scientist for 4 years at Society for Applied Microwave Electronics Engineering and Research (SAMEER), Ministry of IT, Govt. of India, IIT Bombay and worked on various ambitious projects of national relevance. He co-founded www.projectlabs.in where he publishes technical video tutorials for electronics enthusiasts. He has also published several research papers in national, international journals and conferences. He has half decade of teaching experience. Currently he is Assistant Professor at Shri Ramdeobaba College of Engineering and Management, Nagpur.