**Journal: Best Coding Practices**

Justin Starr

Department of STEM

CS 350 – Emerging Systems Architecture & Technologies

Professor Mike Prasad

February 4, 2024

**Journal: Best Coding Practices**

The following is a short analysis of best coding practices and common pitfalls in Embedded C. Best practices while coding in Embedded C include writing clean and readable code, applying efficient memory management techniques, applying debugging and error handling strategies, utilizing effective testing and verification practices, creating safe and secure code, and applying power optimization techniques (Indian Institute of Embedded Systems (IIES), n.d.).

Writing clean and readable code includes using appropriate naming conventions that are understandable, using empty lines to help keep code organized and readable, never passing more than three parameters to a function, using four spaces instead of the tab key, adding spaces after keywords and no space between function names and parenthesis, using in-line comments to explain code and restrict lines to no more than 80 characters (Indian Institute of Embedded Systems (IIES), n.d.). Efficient memory management techniques include Segmentation which involves breaking up physical memory into various chunks, paging, which is a way of organizing memory into uniformly sized chunks, and swapping, which is a mechanism for managing memory space effectively (Indian Institute of Embedded Systems (IIES), n.d.). Applying debugging and error handling strategies include using force to debug, introduction and deduction technique, using a backtracking method and testing to debug, programming with care and placing controls within your code, handling exceptions, writing verification code and test cases, and in the end debugging your code (Indian Institute of Embedded Systems (IIES), n.d.). Re-examining functional and non-functional characteristics and requirements helps ensure the finished product is free of flaws during the testing and verification process (Indian Institute of Embedded Systems (IIES), n.d.). Safety and security considerations include using secure network communication protocols, employing firewalls, data encryption, authentication, and hardware-assisted control-flow monitoring (Indian Institute of Embedded Systems (IIES), n.d.). Lastly, techniques for power optimization include placing case labels in a narrow range and frequent case labels first, minimizing the use of local variables, reducing the number of parameters, preferring int over char, use profile guided optimization, using hardware accelerators and SIMD hardware, avoiding cascaded function calls, defining lightweight constructors, and preferring initialization over an assignment (Indian Institute of Embedded Systems (IIES), n.d.).

Common pitfalls in Embedded C include Mixing signed and unsigned integers in arithmetic operations, Overstepping array boundaries, missing out the base condition in recursive function, using character constants instead of string literals and vice versa, by default, floating point literals are of type double, forgetting to free memory, forgetting semi-colons, writing = instead of == when making comparisons, and copying too much (NerdyElectronics, n.d.).

Due to arithmetic conversions that occur before comparisons, mixing signed and unsigned integers in arithmetic operations results in erroneous results (NerdyElectronics, n.d.). Overstepping array boundaries could altogether corrupt the buffer or cause segmentation fault by accessing different memory locations. Because floating point literals are of type double by default, mixing float variables with double literals will result in poor performance on platforms that do not have the hardware the support for double precision (NerdyElectronics, n.d.). Forgetting to free memory could cause a memory leak that could accumulate substantially as wasted memory which is unavailable to you or the system, and ultimately it could cause unexpected behavior and crashes. The ‘=’ operator is used to assign with and the ‘==’ operator is used for comparisons. If the ‘=’ operator is used when it should have been ‘==’, incorrect variable assignments can be made when comparisons are supposed to be checked. When it comes to copying too much, it is important that we allocate sufficient space for data. Not doing so can lead to buffer overflow which is often exploited by malicious attackers to cause a program to crash, corrupt data, and even steal data (Preforce Software, Inc., n.d.).

**References**

Indian Institute of Embedded Systems (IIES). (n.d.). Tips and Best Practices for Embedded C

Programming. <https://iies.in/blog/embedded-c-programming-tips-and-best-practices/>

NerdyElectronics. (n.d.). Most common pitfalls in C Programming language and how to avoid

them. <https://nerdyelectronics.com/most-common-pitfalls-in-c/>

Perforce Software, Inc. (n.d.). How to Write Secure Code in C.

<https://www.perforce.com/sites/default/files/pdfs/how-to-write-secure-code-c.pdf>