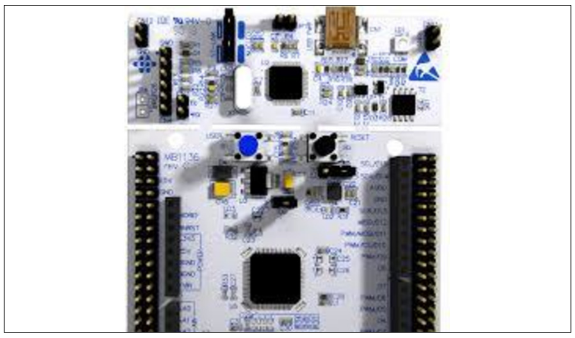
Marc Benz

Samuel Niederer

Students	Marc Benz, Samuel Niederer
Examiner	Prof. Reto Bonderer
Subject Area	Embedded Software Engineering

C++Exception Handling vs Return Codes

C++Exception Handling vs Return Codes on Embedded Systems


Nucleo-64 development board
ST-Microelectronics, UM1724 User manual

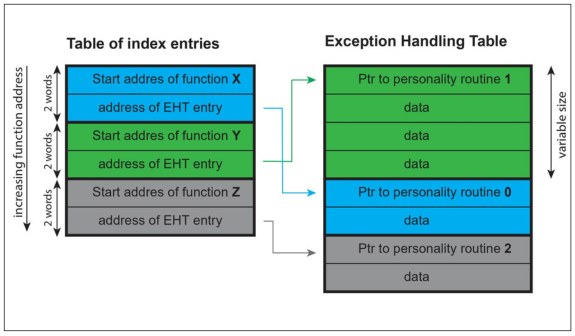
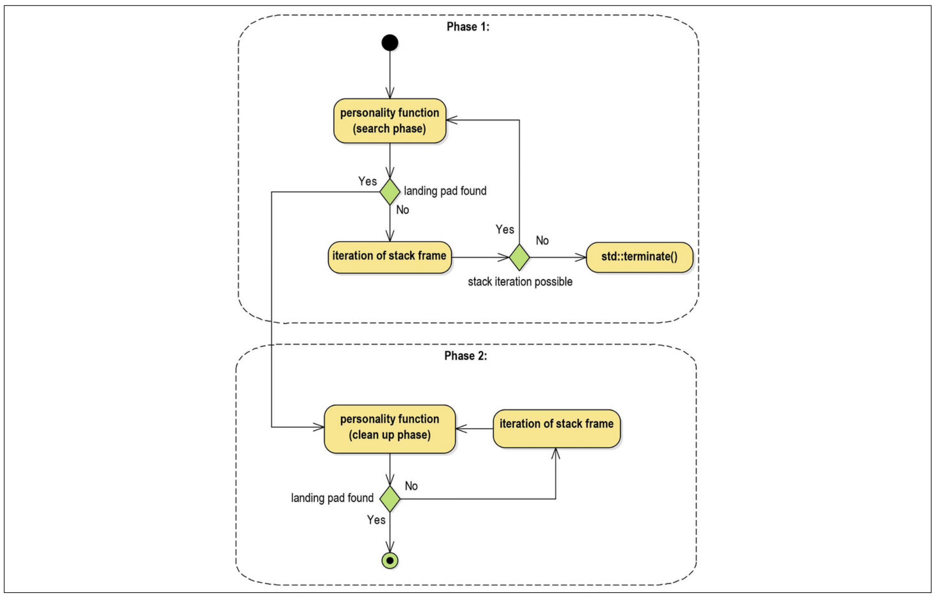


Table of index entries
Exception Handling Table
increasing function address
2 words
2 words
2 words
Start addres of function X
address of EHT entry
Start addres of function Y
address of EHT entry
Start addres of function Z
address of EHT entry
Ptr to personality routine 1
data
data
data
Ptr to personality routine 0
data
Ptr to personality routine 2
data
variable size

Illustration of the Exception Handling Table
Own presentment



Activity diagram of stack unwinding
Own presentment

Introduction: In computer programming there are situations during run time which are exceptional and need special treatment. The response to the occurrence of such an exceptional situation is known as error handling. There are two common ways of dealing with such events, error codes and exception handling. Error codes are said to be messy and unreadable, whereas exception handling has the reputation of being slow and heavy. The bad reputation of C++ exception handling stems mainly from its early days, when it was implemented in such a way that it slowed down the execution time even if no error occurred. A lot has changed since the first implementation of exception handling in 1998, nevertheless the adoption on embedded systems is very small to this day.

Objective: The objective of this paper is the examination of the differences in performance and readability between error codes and exception handling on embedded systems. For our analysis we used the widespread Cortex M4 microcontroller from the established ARM Cortex family. A mixture of available debugging tools in combination with self-developed software enabled it to analyse code fragments tailored to our questions.

Conclusion: Our measurements showed that exception handling is still not the one and only solution for C++ programs on embedded systems. However, it can be a clean and safe solution if it is used in the right way. It is time to forget about the old legends and learn about the new implementations with their own benefits and drawbacks. As researchers work on new implementations, there is a chance of upcoming solutions that could solve some of the problems exception handling still has left. Perhaps we will see a wider adoption on embedded systems if they turn out to be viable.