TABLE OF CONTENTS

Introduction:

In the fast-paced and ever-evolving world of technology, the need for efficient and powerful operating systems has become paramount. The advent of 64-bit computing has revolutionized the way we interact with our digital devices, enabling a new era of speed, performance, and scalability. Intel processors have been evolving from 4-bit (Intel 4004) to 64-bit (i5, i7, i9).

The labels "64-bit," 32-bit," etc. designate the number of bits that each of the processor's general-purpose registers (GPRs) can hold. So when someone uses the term "64-bit processor," what they mean is "a processor with GPRs that store 64-bit numbers." And in the same vein, a "64-bit instruction" is an instruction that operates on 64-bit numbers (Stokes).

64-bit processors were designed for gaming purposes, for using 3D design programs, architectural programs and all the other programs that require more processing horsepower than the 32-bit. With HD becoming the new standard and everyday-use programs becoming more involved and multimedia friendly, the 64 bit processor is becoming more commonplace (Russell).

There are two different types of 64-bit architectures an administrator might encounter: x86 and IA64. The most used is x86 (Orchilles). And in this report x86_64 bit processors are reviewed.

Internal Architecture:

Since the 64-bit registers allow access for many sizes and locations, we define a byte as 8 bits, a word as 16 bits, a double word as 32 bits, a quadword as 64 bits, and a double quadword as 128 bits. Intel stores bytes “little endian,” meaning lower significant bytes are stored in lower memory addresses (Intel).

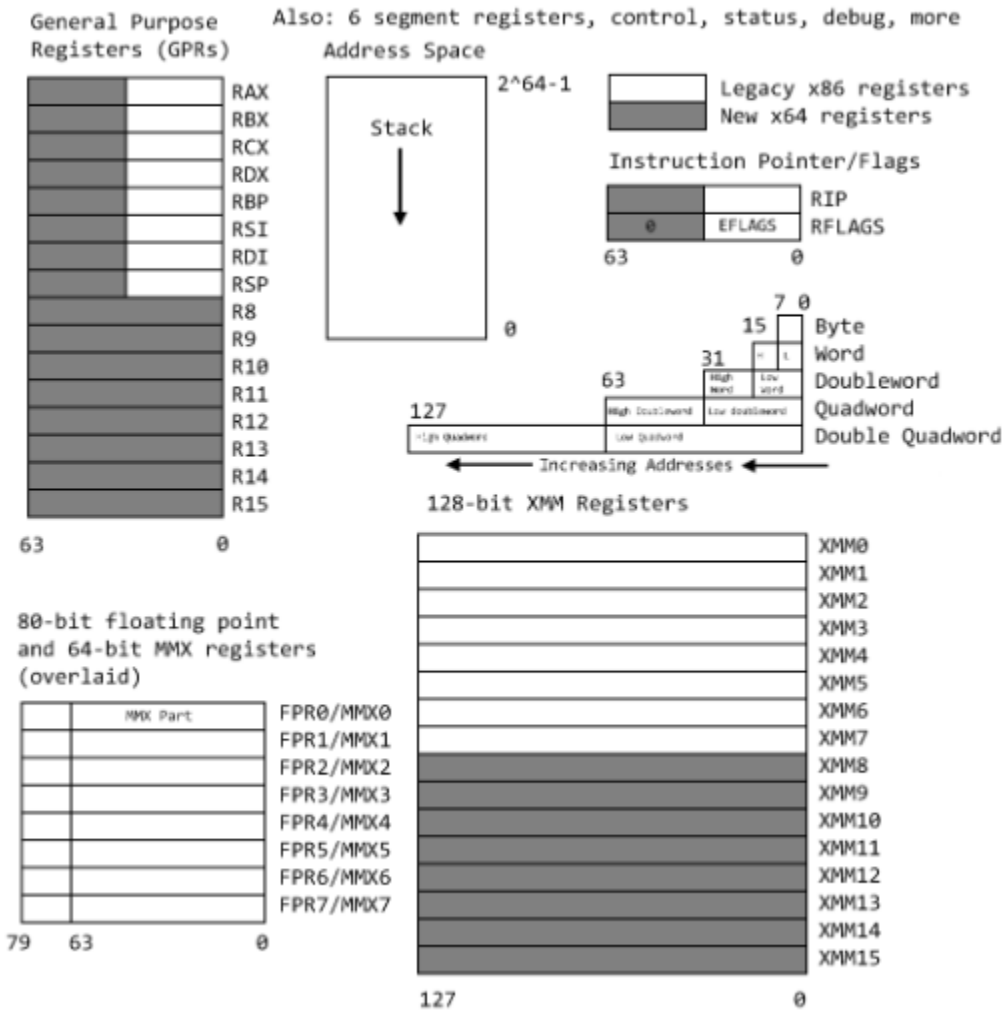

Fig 1- General Architecture

1. **Registers**: The x64 architecture introduces additional general-purpose registers compared to the 32-bit x86 architecture. It has 16 general-purpose registers, each 64 bits wide, named RAX, RBX, RCX, RDX, RSI, RDI, RBP, RSP, R8-R15. These registers can be used for various purposes such as storing data, addresses, or intermediate results during computations. The 64-bit instruction pointer RIP points to the next instruction to be executed (Intel).

2. **Addressing**: The x64 architecture supports a larger virtual address space of up to 2^64 bytes (16 exabytes), compared to the 32-bit x86 architecture's 4 GB limit (“Intel® 64 and IA-32 Architectures Developer's Manual: Vol. 3A”). This allows applications to access and manipulate significantly more memory. The extended address space enables larger datasets, better support for complex computations, and improved performance for memory-intensive tasks.

3. **Instruction Set**: The x64 architecture retains backward compatibility with the x86 instruction set, which means that 32-bit x86 software can still run on x64 systems. However, it introduces new instructions that take advantage of the 64-bit registers and enhanced capabilities. These new instructions enable more efficient memory access, enhanced floating-point operations, and support for advanced features like SIMD (Single Instruction, Multiple Data) for parallel processing.

4. **Memory Management**: The x64 architecture employs a paging-based memory management system, similar to its 32-bit predecessor. It utilizes a hierarchical page table structure to map virtual addresses to physical addresses. The larger address space allows for a larger number of pages and more efficient memory management, resulting in improved performance and the ability to handle larger datasets.

5. **Exception and Interrupt Handling**: The x64 architecture maintains the exception and interrupt handling mechanism from the x86 architecture. It provides a dedicated interrupt descriptor table (IDT) that contains interrupt and exception handlers. When an interrupt or exception occurs, the processor transfers control to

the appropriate handler, which can be either a hardware interrupt (e.g., keyboard input) or a software exception (e.g., division by zero).

6. **Execution Modes**: The x64 architecture supports multiple execution modes, including the long mode, which is the primary mode for 64-bit operating systems and applications. It also supports compatibility modes, legacy mode where the CPU behaves like a 32-bit CPU and all the 64-bit enhancements are turned off (Perla).

7. **System Components**: An x64 system typically consists of a central processing unit (CPU) that implements the x64 architecture, one or more memory modules, storage devices, and various input/output (I/O) devices. The CPU interacts with these components through buses and controllers, allowing data transfer and communication between different parts of the system.

8. **Stack**: The x64 architecture utilizes a stack to store temporary data, function call information, and local variables. The stack pointer RSP points to the last item pushed onto the stack, which grows toward lower addresses (Intel). The stack is an essential component for managing function calls and maintaining the execution context.

9. **SIMD and Floating-Point Operations**: The x64 architecture includes an expanded set of registers dedicated to SIMD operations, known as XMM registers. These registers support Single Instruction, Multiple Data (SIMD) operations, allowing for parallel processing and efficient execution of tasks involving multimedia, graphics, and scientific computations. The architecture includes registers for floating-point operations, such as the X87 floating-point stack and SSE (Streaming SIMD Extensions) registers, which enable efficient handling of floating-point calculations.

10. **System-Level Protection**: To ensure system security and stability, the x64 architecture provides various protection mechanisms. These include privilege levels and rings, which separate and restrict access to system resources based on privilege levels. The architecture supports multiple rings (often referred to as ring 0, ring 1, etc.), with lower-numbered rings having higher privileges. This protection mechanism helps isolate and protect critical system components from unauthorized access or malicious software.

11. **Caching and Memory Hierarchy**: x64 systems employ a multi-level cache hierarchy. Data caches usually are organized in hierarchies of between 1 and 3 levels. (“Processor Microarchitecture - An Implementation Perspective”)

    These caches are located closer to the CPU and provide faster access to frequently used data and instructions. Caches help reduce memory latency by storing copies of recently accessed data and instructions.

12. **Memory Ordering and Barrier Instructions**: The x64 architecture includes memory ordering and barrier instructions to control the order of memory operations. These instructions ensure that memory reads and writes are performed in the expected sequence, preventing potential issues like data races or inconsistent values. Memory barrier instructions provide synchronization points, allowing for coordination between multiple threads or processes.

13. **Instruction Pipelining and Out-of-Order Execution**: To improve instruction throughput and overall performance, x64 processors employ advanced techniques such as instruction pipelining and out-of-order execution. Instruction pipelining breaks down instructions into smaller stages and allows simultaneous execution of multiple instructions. Out-of-order execution reorders instructions dynamically to maximize the utilization of execution units and minimize idle time, resulting in improved performance.

64-bit vs 32-bit:

Aspects	64-bit System	32-bit System
Registers	Has 16 general-purpose registers, each 64 bits wide	Has 8 general-purpose registers, each 32 bits wide
Addressing	Utilizes a 64-bit address space for larger memory access	Utilizes a 32-bit address space for memory access
Instruction Set	Supports an expanded instruction set with 64-bit instructions	Supports a 32-bit instruction set
Memory Paging	Has 4-level paging mode	Has no 4-level paging mode
Floating-Point Unit	Provides enhanced support for 64-bit floating-point calculations	Supports 32-bit floating-point calculations
SIMD Operations	Supports larger registers and enhanced SIMD instructions	Supports limited SIMD capabilities with smaller registers
Compatibility	Supports both 64-bit and 32-bit software applications	Primarily supports 32-bit software applications
Cache Hierarchy	Utilizes larger cache sizes for improved performance	Typically has smaller cache sizes

```c
int a, b, c, d, e;
for (a = 0; a < 100; a++) {
    b = a;
    c = b;
    d = c;
    e = d;
}
```

A 64-bit system has more GPRs than a 32-bit system, so a significant speed increase for tight loops since the processor does not have to fetch data from the cache or main memory if the data can fit in the available registers.

The code aside is written in C. The code demonstrates the data movement from register to register if available else it will have to store in memory and again fetch for use.

Limitations:

1. **Memory Overhead**: 64-bit systems consume more memory due to longer pointers and larger data types, potentially impacting memory usage and cache efficiency.
2. **Compatibility Issues**: Some older software or drivers may not be compatible with 64-bit architecture, limiting the usability of certain programs or devices.
3. **Physical Address Space Limitations**: Hardware configurations and operating systems can impose limitations on the amount of physical memory that can be addressed, even on 64-bit systems.
4. **Process Address Space Allocation**: Operating systems may reserve portions of the process address space for the kernel, reducing the available address space for user applications.
5. **Memory-Mapped Files**: Handling files larger than 4 GB can be challenging on 32-bit architectures, as only parts of the file can be mapped into the address space at a time, impacting the efficiency of memory-mapped file operations.

References:

Intel. *Introduction to x64 assembly*. www.intel.com, https://www.intel.com/content/dam/develop/external/us/en/documents/introduction-to-x64-assembly-181178.pdf .

"Intel® 64 and IA-32 Architectures Developer's Manual: Vol. 3A." *Intel*, https://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-vol-3a-part-1-manual.html .

"Processor Microarchitecture - An Implementation Perspective." *UCSD CSE*, https://cseweb.ucsd.edu/classes/fa14/cse240A-a/pdf/04/Gonzalez_Processor_Microarchitecture_2010_Claypool.pdf .

Russell, Sean. "List of 64-Bit Processors." *Techwalla*, https://www.techwalla.com/articles/list-of-64-bit-processors

Stokes, Jon. "An Introduction to 64-bit Computing and x86-64." *Ars Technica*, 11 March 2002, https://arstechnica.com/gadgets/2002/03/an-introduction-to-64-bit-computing-and-x86-64 .

Orchilles, Jorge. "Microsoft Windows 7 Administrator's Reference". 2010.

Perla, Enrico. "A Guide to Kernel Exploitation". 2011.