Execute Linux from 256/512 Mb HyperRAM

About this document

Scope and purpose

This application note discusses SDRAM-less Linux systems in detail that use just HyperRAM™ as main memory. The document highlights the pros and cons of the required RAM footprint optimizations, and provides solutions and workarounds. In addition, this application note also describes a manually optimized implementation based on Xilinx UltraScale+, with its performance results at a system level.

Table of contents

1 Introduction

Embedded Linux systems usually include a large amount of SDRAM (in the order of several gigabytes). Not all Linux systems actually require that much RAM. For smaller systems with lower requirements such as Wi-Fi routers or IoT web services, it can be interesting to replace the classic SDRAM by serial memory such as HyperRAM™. Several advantages come with this approach.

First of all, the classic SDRAM bus interface requires a high pin count with separate address and data lines complemented by another set of control signals. By using multiplexed serial interfaces such as HyperBus™, this pin count can be reduced significantly. Eventually, PCB complexity decreases and with that also the total system cost.

In addition, from a performance point of view, modern high-speed serial busses such as HyperBus or Octal SPI have become very powerful. Performance is now sufficient to be used for smaller Linux applications that do not require a full-blown graphical windows system. In this case, a comparable user experience can be achieved compared to the classic SDRAM.

2 32-bit vs. 64-bit Linux kernels

The HyperRAM family includes packages with higher densities of 256 Mbit and 512 Mbit. Although HyperRAM densities have increased, it should be noted that they are still lower than those of commodity SDRAM parts. Therefore, when used for HyperRAM-based Linux systems, RAM space is usually scarce; thus, optimizing the memory footprint is important.

One step towards this goal is to choose a 32-bit kernel even if the hardware supports 64-bit addressing. If there is no need to address more than 4 GB including I/O devices, a 32-bit Linux kernel is sufficient. This setup is widely used for embedded Linux systems such as on the Raspberry Pi platform in order to save resources.

If a 64-bit kernel must be used, special optimizations become necessary to reduce its usually enormous RAM requirements. A critical element in this case is the page table that maps virtual addresses to physical addresses. On 64-bit Linux systems, this table can easily consume multiple MBs of RAM. However, the page table can be optimized to cover only a subset of all theoretically addressable pages. This significantly reduces its footprint.

3 Choosing the right Linux distribution

The simplest approach for a small RAM footprint probably is the use of already optimized Linux distributions. There are special distributions in the market that have been optimized in terms of RAM usage and that can be used to build lightweight embedded systems. For example, **OpenWrt Linux** that targets networking and routing devices already includes a vast set of memory footprint optimizations. Many components have been optimized to be small enough to fit into the limited storage and memory available in networking systems. This is an easy solution and is preferable whenever a suitable distribution is available for the underlying hardware platform.

For example, the first version of the famous **WRT54** Wi-Fi router that was released in 2003 used 16 MB of RAM. Due to strong and efficient software optimizations, later versions could reduce this amount to just 8 MB. Of course, since these days, Linux kernels have increased in complexity and size as well as in their RAM footprint.

4 The manual approach

If no optimized Linux distribution is available for the underlying hardware platform, a set of optimizations must be worked out and applied manually. The following section discusses such an example where a Xilinx ZCU102 UltraScale+ evaluation board (Rev. 1.1) together with the corresponding PetaLinux BSP software is used as the hardware and software basis.

The standard boot configuration of the ZCU102 board starts the system from an external SD or MMC card. On the first boot step, the SoC ROM code loads the first stage boot loader (FSBL) from SD/MMC into the internal SoC SRAM. Afterwards, an enhanced FSBL sets up the required HyperBus memory controller in the Programmable Logic (PL) portion of the SoC so that HyperRAM devices can be accessed. With this enhancement in place, 64 MB (512 Mbit) of HyperRAM can be used to replace the classic DDR4 SDRAM module.

By default, PetaLinux packages the root file system into the INITRAMDISK image and loads it as a RAMFS into RAM before the kernel is eventually started. Approximately 55 MB can be saved by moving the root file system from RAM to the external SD/MMC card. The exact value depends on the packages that are included in the file system. In general, it is a good idea to move the root file system to external storage whenever RAM space is tight.

All software components of the second stage in the boot process (u-boot and Linux kernel) must be modified to use HyperRAM instead of SDRAM. In particular, the situation that there is no physical RAM located anymore at system address 0 is challenging. Things like the interrupt vector table and other components must be relocated to the corresponding base address of the HyperRAM window. Furthermore, to achieve a sufficient performance level, it is crucial to set the right memory attributes (“executable” and “cached”) for the HyperRAM region.

In addition, in the Linux kernel’s memory management code, the size of the system page table must be reduced. By default, the resulting page table is quite large (approximately 64 MB). This is due to the 64-bit setup of the PetaLinux kernel and its capability to address 256 TB of memory directly via pointers. By refining the granularity of the (sparse) page table for HyperRAM, its footprint can be reduced to just 4 MB.

After the boot process, approximately 10 MB of HyperRAM are finally available for user-level applications such as for a lightweight web server or text mode apps.

Total system boot time from flipping the power switch until the login prompt is measured as follows:

- Using DDR4 SDRAM: 16.6 s
- Using HyperRAM: 23.7 s

These results show that the impact of serial HyperRAM on the system boot time is not very large. In fact, 64 MB of HyperRAM can be used pretty well as a replacement for SDRAM on this optimized Linux system if user-level memory requirements are low.

5 Conclusion

Thanks to the high performance of today's serial memory busses, HyperRAM has become a viable alternative to classic SDRAM for small embedded system. The 256/512 Mb densities push the envelope further and enlarge the range of covered Linux applications. For systems with low memory requirements, HyperRAM can help reduce pin count, PCB space and finally overall system cost.

References

[1] **ZCU102 Evaluation Board User Guide**: Describes the Xilinx ZCU102 board based on the Zynq UltraScale+ SoC, its features, specs, components and connectors

Revision history

Document version	Date of release	Description of changes
**	2021-03-26	Initial version

Trademarks
All referenced product or service names and trademarks are the property of their respective owners.

Edition 2021-03-26

Published by

Infineon Technologies AG

81726 Munich, Germany

Do you have a question about this document?

Go to www.cypress.com/support

Document reference
002-31852 Rev. **

IMPORTANT NOTICE

The information contained in this application note is given as a hint for the implementation of the product only and shall in no event be regarded as a description or warranty of a certain functionality, condition or quality of the product. Before implementation of the product, the recipient of this application note must verify any function and other technical information given herein in the real application. Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this application note.

The data contained in this document is exclusively intended for technically trained staff. It is the responsibility of customer's technical departments to evaluate the suitability of the product for the intended application and the completeness of the product information given in this document with respect to such application.

For further information on the product, technology, delivery terms and conditions and prices please contact your nearest Infineon Technologies office (**www.infineon.com**).

WARNINGS

Due to technical requirements products may contain dangerous substances. For information on the types in question please contact your nearest Infineon Technologies office.

Except as otherwise explicitly approved by Infineon Technologies in a written document signed by authorized representatives of Infineon Technologies, Infineon Technologies' products may not be used in any applications where a failure of the product or any consequences of the use thereof can reasonably be expected to result in personal injury.