Методические указания к выполнению лабораторной работы "Демонстрация виртуализации на процессоре Байкал-Т1 с помощью prplHypervisor"

Оглавление

Лабораторная работа №2. Демонстрация виртуализации на процессоре Байкал-Т1 с помощью prplHypervisor

Цель работы: сборка и запуск образа гипервизора prplHypervisor на процессоре Байкал-Т1, изучение возможностей работы prplHypervisor.

Введение

Гипервизор — программа, обеспечивающая одновременное параллельное выполнение нескольких операционных систем (ОС) на одном и том же компьютере (хост-компьютере). Гипервизор обеспечивает изоляцию операционных систем друг от друга, защиту и безопасность, разделение ресурсов между различными запущенными ОС и управление ресурсами.

Гипервизор может предоставлять работающим под его управлением ОС средства взаимодействия между собой (например, через обмен файлами или сетевые соединения) так, как если бы эти ОС выполнялись на разных физических компьютерах.

Гипервизор предоставляет запущенным под его управлением операционным системам службу виртуальной машины, виртуализируя или эмулируя реальное аппаратное обеспечение . Гипервизор позволяет независимое «включение», перезагрузку, «выключение» любой из виртуальных машин, при этом операционная система, работающая в виртуальной машине под управлением гипервизора, может, но не обязана «знать», что она выполняется в виртуальной машине, а не на реальном аппаратном обеспечении.

В данной лабораторной работе предлагается знакомство с гипервизором prplHypervisor™, проектом с открытым кодом с prpl Foundation лицензией. Идея создателей гипервизора prplHypervisor™ - объединить между собой устройства "интернета вещей".

Гипервизор обладает небольшим объемом кода, что обеспечивает ему объем бинарного кода в пределах 30 КБ на флеш памяти и 4 КБ – в оперативной памяти. С помощью prplHypervisor™ можно запускать до шести гостевых систем. Тесты производительности показали минимальные задержки на переключения контекста между гостевыми системами.

Поддержка процессора Байкал-Т1 в данный проект была добавлена Embedded System Group (GSE) 04 апреля 2017 года.

Задание 1. Сборка образа гипервизора prplHypervisor

Получение исходных кодов

Для выполенния данной работы необходимо выполнить этап 1 лабораторной работы №1 (установка и сборка SDK Байкал-Т1).

Для выполнения данной работы необходимо скопировать в домашнюю директорию предварительно скачанный с GitHub репозиторий, который расположен на сервере ЛэБ в директории `/srv/sdk/lab2/prpl-hypervisor.tar.gz`.

Основное дерево разработки гипервизора prplHypervisor находится на GitHub по адресу: `https://github.com/prplfoundation/prpl-hypervisor.git`

Чтобы распаковать архив, находясь в домашней директории, наберите следующую команду:

```
$ tar xvfz prpl-hypervisor.tar.gz
```

Сборка гипервизора

Для сборки гипервизора под процессор Байкал-Т1 на сервере ЛэБ нужно в Makefile из директории

`<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/` поменять путь до кросс-компилятора.

Пример правильного пути:

```
# CROSS Compiler
CROSS_COMPILER = /srv/sdk/baikal/usr/x-tools/mipsel-unknown-linux-gnu
                                                    /bin/mipsel-unknown-linux-gnu-
```

После выполненных действий можно приступать к сборке образа гипервизора. Для этого нужно выполнить команду `make` в директории

`<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/`

Пример результата успешной сборки приведен ниже.

```
gcc -DBAIKAL_T1 -o genconf ../cfg_reader/genconf.c -lconfig
#execute first and exit in case of errors
./genconf cfg/sample-ping-pong.cfg baikal-t1 || (exit 1)
ping pong
#execute and export to a makefile variable the ouput of the script
mipsel-unknown-linux-gnu-gcc -DCPU_ID="P5600" -DCPU_ARCH="Baikal-
T1 board." -DCPU_FREQ=850000000 -EL -O2 -mips32r2 -Wall -Wa,-mvirt
-mno-check-zero-division -msoft-float -fshort-double  -c
```

```
-ffreestanding -nostdlib -fomit-frame-pointer -G 0
-DHYPVERSION="v0.12.111 (g662ee0a)" -DBAIKAL_T1
-I../../arch/mips/baikal-t1/include
-I../../arch/mips/common/include -I../..//platform/include
-I../../sys/lib/include -I../../sys/kernel/include
-I../../platform/baikal_t1_board/include -I../..//include \
        ../../sys/lib/libc.c \
        ../../sys/lib/linkedlist.c \
        ../../sys/lib/malloc.c \
        ../../sys/lib/queue.c
...
mipsel-unknown-linux-gnu-size ../../apps/pong/pong.elf
   text    data     bss     dec     hex filename
  15396      16    3472   18884    49c4 ../../apps/pong/pong.elf
make[1]: Leaving directory '/home/sumatra/hyper/prpl-
hypervisor/bare-metal-apps/platform/baikal_t1_board'
0+1 записей получено
1+0 записей отправлено
65536 байт (66 kB, 64 KiB) скопирован, 0,00199796 s, 32,8 MB/s
0+1 записей получено
1+0 записей отправлено
65536 байт (66 kB, 64 KiB) скопирован, 0,000421903 s, 155 MB/s
```

Задание 2. Запуск образа гипервизора prplHypervisor

Гипервизор позволяет запускать до 6 bare-metal приложений (виртуальных машин). Размер SRAM для каждой машины может быть изменен.

В директории `<PATH_TO_PRPL_HYPERVISOR>/bare-metal-apps/apps` расположены папки: `blink`, `ping`, `pong` и другие. В каждой папке находится исходный код bare-metal приложения. В процессе виртуализации эти приложения становятся гостевыми ОС.

Конфигурация prplHypervisor задается с помощью текстовых файлов `.cfg`. В процессе компиляции при чтении конфигурационного файла создается заголовочный файл `config.h` в директории платформы. В директории

`<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/cfg` находится несколько конфигурационных файлов.

В cfg-файлах имеется две части: конфигурация системы и конфигурация виртуальных машин.

Пример части, описывающей конфигурацию системы:

```
system = {
   debug =  [ "WARNINGS", "INFOS", "ERRORS"];
   uart_speed = 115200;
```

```
   scheduler_quantum_ms = 10;
   guest_quantum_ms = 1;
};
```

Переменная `scheduler_quantum_ms` определяет время переключения между гостевыми ОС в милисекундах. Значение 10 в примере указывает, что гостевая ОС будет работать в течение 10 мс до того, как управление будет передано другой гостевой ОС. Для эмуляции прерываний таймера в гостевых ОС гипервизор посылает виртуальные прерывания гостевым ОС с интервалом, равным `guest_quantum_ms` (в милисекундах).

Пример части, описывающей конфигурацию виртуальных машин:

```
virtual_machines = (
    {
        app_name = "ping";
        os_type = "BARE_METAL";
        priority = 100;
        RAM_size_bytes = "MEM_SIZE_32KB";
    },
    {
        app_name = "pong";
        os_type = "BARE_METAL";
        priority = 100;
        RAM_size_bytes = "MEM_SIZE_32KB";
    }
);
```

Переменные, используемые при описании виртуальных ОС:

- `app_name` - название приложения;
- `os_type` - тип гостевой ОС (допустимое значение - `BARE_METAL`);
- `priority` - приоритет (допустимые значения: от 0 до 255);
- `RAM_size_bytes` - размер оперативной памяти виртуальной ОС, допустимые значения:
  - `MEM_SIZE_32KB`,
  - `MEM_SIZE_64KB`,
  - `MEM_SIZE_128KB`,
  - `MEM_SIZE_256KB`;

Поддержка Байкал-Т1 включает в себя прерывания, таймер и драйвер для UART, поддержка остальных аппаратных компонентов отсутствует.

Рассмотрим пример запуска 2-х виртуальных машин - программу обмена сообщениями ping-pong. Она представляет собой пример обмена сообщениями между гостевыми ОС. Первая виртуальная машина (ping) посылает сообщение второй (pong) и получает ответ.

Для запуска этого примера в Makefile из директории

`<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/` задан конфигурационный файл

`CFG_FILE = cfg/sample-ping-pong.cfg`. Следует убедиться, что это так. Если нет - нужно указать данный файл и заново осуществить сборку образа гипервизора.

Запуск гипервизора на Байкал-Т1 будет осуществляться через TFTP-протокол.

Скопируйте файл `firmware.bin` из директории

`<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/` в директорию `srv/tftp/<YOUR_CARD>` на TFTP-сервере. После этого нужно с помощью комнады `ls -l` проверить, что права доступа загруженного файла соответсвуют 755.

Для запуска гипервизора после загрузки меню u-boot необходимо перейти в командную строку путем выбора пункта меню "U-Boot console"(см. рисунок 1). Для перехода в меню загрузки необходимо сразу после включения питания платы нажать клавишу `<S>` (только для платы miniATX).

Необходимо убедиться, что хост-машина и загрузчик находятся в одной подсети, если это не так, то произведите соответствующую настройку согласно описанию в приложении. Для уточнения IP-адреса tftp-сервера необходимо обратиться к документу, описывающему правила работы с платами, который представлен на сайте проекта.

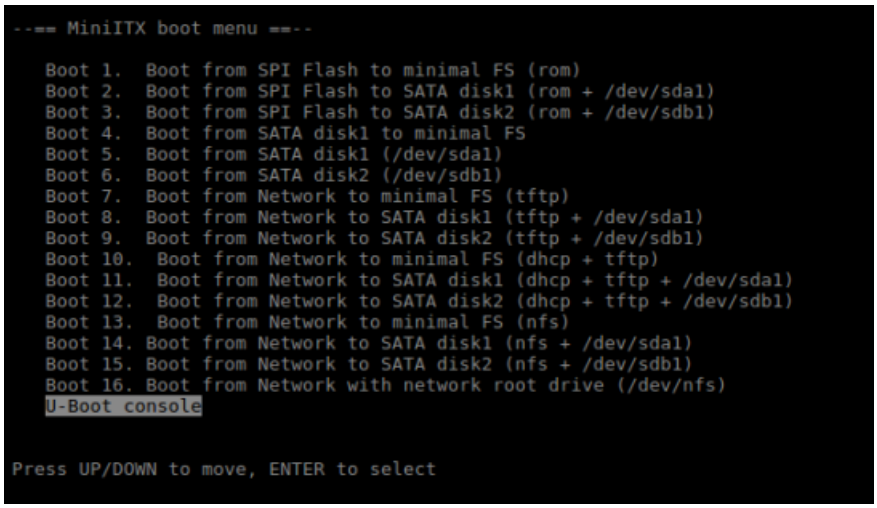

Рис. 1 Выбор консоли u-boot

После входа в консоль u-boot загрузите данный файл на плату с помощью команды:

```
BAIKAL #tftp 0xa0000000 192.168.68.1:firmware.bin
```

Обратите внимание, что ip-адрес в приведенном примере команды следует заменить на ip-адрес вашего TFTP-сервера.

В результате успешной загрузки будет выведено сообщение:

```
dwmac.bf05e000 Waiting for PHY auto negotiation to
complete......... TIMEOUT !
dwmac.bf05e000: No link.
Speed: 100, full duplex
Using dwmac.bf060000 device
TFTP from server 192.168.68.1; our IP address is 192.168.68.230
Filename 'firmware.bin'.
Load address: 0xa0000000
Loading: ###
         381.8 KiB/s
done
Bytes transferred = 131072 (20000 hex)
```

Запуск гипервизора осуществляется с помощью команды:

```
BAIKAL #go 0x80002000
```

Результат успешного запуска приведен ниже.

```
## Starting application at 0x80002000 ...
============================================================
prplHypervsior . () [Sep 12 2017, 17:29:31]
Copyright (c) 2016, prpl Foundation
============================================================
CPU Core:       P5600
Board:          Baikal-T1 board.
System Clock:   1200MHz
Heap Size:      41Kbytes
Scheduler:      10ms
Guest Tick:     1ms
VMs:            2

Initializing Virtual Machines.
Configuring ping VM starting at 0x80010000 RAM address.
Configuring pong VM starting at 0x80018000 RAM address.
P5600 core in Vectored Interrupt Mode.
Inter-VM communication hypercalls registered.
UART driver enabled.
```

```
Starting hypervisor execution.

Measuring Inter VM communication latency.

Target VCPU not initialized.
Target VCPU not initialized.

Wait...

Round trip latency for messages 64 bytes long.

Average 2029.46 us
Worst 2861 us
Best 2818 us

One way latency for messages 64 bytes long.

Average 14801.86 us
Worst 47371 us
Best 1406 us
...
```

Рассмотрим, как запустить другое приложение из имеющихся примеров. Предлагается запустить приложение `blink`, осуществляющее вывод на экран значений счетчика.

Для этого в Makefile из директории `<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/` нужно поменять конфигурационный файл на `sample-blink.cfg` и осуществить пересборку образа гипервизора.

Для запуска гипервизора необходимо повторить действия, описанные в предыдущем задании.

В результате удачного запуска будет напечатано:

```
============================================================
prplHypervsior v0.12.126 (gb22f550) [Oct 18 2017, 13:11:25]
Copyright (c) 2016, prpl Foundation
============================================================
CPU Core:       P5600
Board:          Baikal-T1 board.
System Clock:   1200MHz
Heap Size:      41Kbytes
Scheduler:      10ms
Guest Tick:     1ms
VMs:            1
```

```
Initializing Virtual Machines.
Configuring blink VM starting at 0x80010000 RAM address.
P5600 core in Vectored Interrupt Mode.
Inter-VM communication hypercalls registered.
UART driver enabled.
Starting hypervisor execution.


Blink LED! Total 0 of timer ticks


Blink LED! Total 1 of timer ticks


Blink LED! Total 2 of timer ticks


Blink LED! Total 3 of timer ticks

...
```

Задание 3. Создание собственного bare-metal приложения и его запуск с помощью гипервизора

Самый простой способ создания нового bare-metal приложения - скопировать папку существующего приложения и изменить ее содержимое.

Реализуем простейшее приложение - печать на экран строчки `Hello, world!`. Для этого нужно создать копию папки приложения `blink` (или любого другого) в директории `<PATH_TO_PRPL_HYPERVISOR>/bare-metal-apps/apps`, переименовать ее и находящиеся в ней файл с исходным кодом в `helloworld` и `helloworld.c` соответственно.

В содержимое файла `helloworld.c` следует записать:

```
#include <arch.h>
#include <libc.h>
#include <platform.h>
#include <io.h>
```



```
int main()
{

printf("\nHello, world!\n");
        return 0;
}
```

В директории `<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/cfg` нужно создать конфигурационный файл `sample-helloworld.cfg` следующего содержания:

```
system = {
    debug =  [ "WARNINGS", "INFOS", "ERRORS"];
    uart_speed = 115200;
    scheduler_quantum_ms = 10;
    guest_quantum_ms = 1;
};


/* Virtual Machines Configuration */
virtual_machines = (
    {
        app_name = "helloworld";
        os_type = "BARE_METAL";
        priority = 100;
        RAM_size_bytes = "MEM_SIZE_64KB";
    }
);
```

После этого в Makefile из директории `<PATH_TO_PRPL_HYPERVISOR>/platform/baikal_t1_board/` нужно поменять конфигурационный файл на `sample-helloworld.cfg` и осуществить пересборку образа гипервизора. Заупск осуществить аналогично предыдущим заданиям.

Результат успешного запуска:

```
============================================================
prplHypervsior v0.12.126 (gb22f550) [Oct 18 2017, 13:30:11]
Copyright (c) 2016, prpl Foundation
============================================================
CPU Core:       P5600
Board:          Baikal-T1 board.
System Clock:   1200MHz
Heap Size:      41Kbytes
Scheduler:      10ms
Guest Tick:     1ms
```

```
VMs:           1

Initializing Virtual Machines.
Configuring helloworld VM starting at 0x80010000 RAM address.
P5600 core in Vectored Interrupt Mode.
Inter-VM communication hypercalls registered.
UART driver enabled.
Starting hypervisor execution.



Hello, world!
```

Далее вам предлагается самостоятельно, используя полученные знания, создать и запустить 6 приложений (максимальное количество для данной реализации гипервизора), каждое из которых выводит на экран сообщение: `Hi! I'm <virtual machine ID> Machine!`

Переменная `<virtual machine ID>` - идентификационный номер виртуальной машины, присваевамый ей гипервизором в процессе ее инициализации. ID соответствует порядку инициализации виртуальных машин. Первая машина получает ID 1, вторая - 2 и т.д. ID используется для идентификации виртуальной машины в процессе ее взаимодействия с другими виртуальными машиными. Виртуальная машина может узнать свой ID с помощью гипервызова `get_guestid()`.

Контрольные вопросы

1. Что такое гипервизор, какие возможности он предоставляет и для чего используется?
2. Какие бывают типы гипервизоров и к какому из них относится prpl-hypervisor?
3. Для чего используются конфигурационные файлы и какие возможности они предоставляют?

Приложение. Подготовка хост-машины к работе

Для успешного соединения между u-boot и tftp-сервером, запущенном на хост-машине, необходимо, чтобы они находились в одной подсети.

Настройка адресов в u-boot

Для изменения адресов в u-boot необходимо воспользоваться командой `setenv`:

```
setenv serverip <ip address>
setenv ipaddr <ip address>
```

Для просмотра параметров используется команда `printenv`.

`ipaddr` отвечает за адрес хоста в сети, а `serverip` отвечает за адрес, по которому пойдет обращение к серверу tftp, то есть в этом поле необходимо указать адрес хост-машины, на которой будет запущен tftp-сервер (также важно, чтобы сервер, работающий на хост-машине, слушал все адреса). Необходимо также произвести соответствующую конфигурацию сетевого интерфейса хост-машины, её адрес должен быть точно таким же, как и адрес, заданный в поле `serverip`.

