FREEBSD ON ARM64 HYPER-V

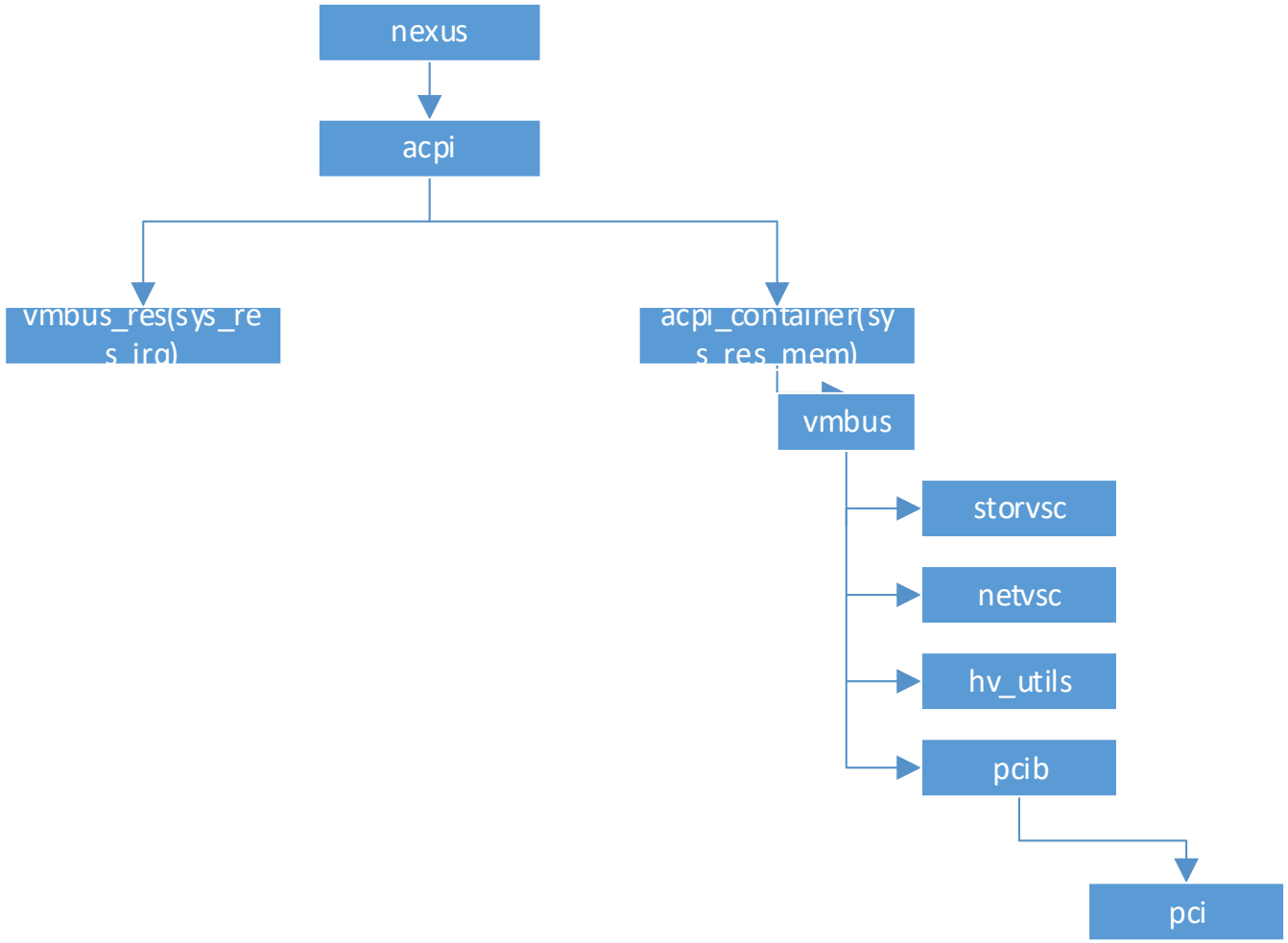
- Wei Hu (whu@microsoft.com)

-Souradeep Chakrabarti(schakrabarti@microsoft.com)

PREFACE

- Microsoft is currently offering Linux in ARM64 SKUs of Azure.
- FreeBSD is available for x86 SKUs in Azure.
- We are working on enabling FreeBSD on ARM64 SKUs of Azure.
- The following slides are on the major changes done to make it happen.

FREEBSD ON ARM64



EXISTING HYPER-V DEVICE DRIVER LAYOUT

- `vmbus/`. The parent of all Hyper-V devices. It also contains code for early initialization, i.e. before any drivers are loaded.
- `vmbus/amd64/ and vmbus/i386/`. Contains vmbus IDT vector entry and hypercall.
- `storvsc/`. Synthetic SCSI controller driver.
- `netvsc/`. Synthetic network controller driver.
- `pcib/`. PCI bridge driver for SR-IOV/pass-through.
- `input/`. Synthetic keyboard driver.
- `utilities/`. Drivers for KVP, VSS, time synchronization etc.
- `include/`. Shared header files; exposed by the vmbus.

NEW HYPER-V DEVICE DRIVER LAYOUT

- vmbus/aarch64/. Contains vmbus_aarch64.c, hyperv_reg.h, hyperv_machdep.h, hyperv_machdep.c, hyperv_aarch64.c.

These files are specific for ARM64 Hyper-V.

- vmbus_aarch64.c : Contains new interrupt handler setup and teardown code.
- hyperv_aarch64.c : Contains Hyper-V identify.
- hyperv_machdep.c : Contains new hypercalls for ARM64 Hyper-V.
- hyperv_reg.h : Contains ARM64 specific synthetic MSR values.

CONTD.

- - vmbus/x86/. Contains vmbus_x86.c, hyperv_x86.c,hyperv_reg.h, hyperv_machdep.h .

These are for both i386 and amd64.

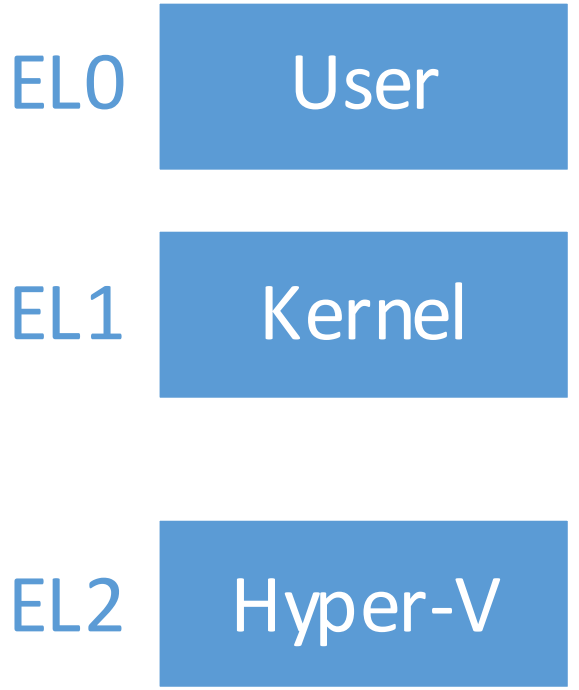
Also new file introduced hyperv_common_reg.h, which contains common

synthetic MSR values for Hyper-V.

This approach to avoid redundancy of the code.

ARM SMCCC HVC

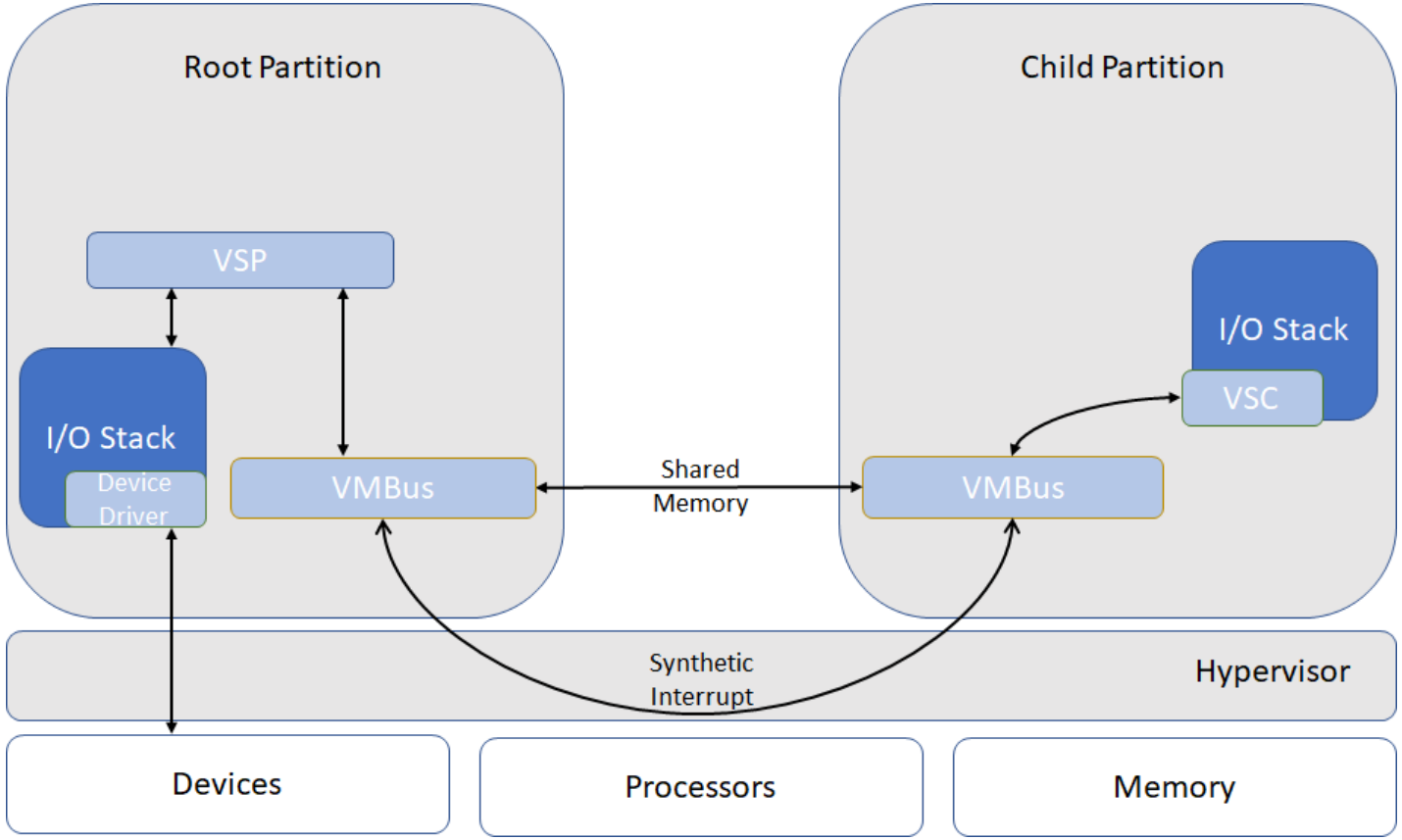
- To implement writing of MSR and reading of MSR in ARM64 HvCallSetVpRegisters hypercall and HvCallGetVpRegisters hypercall is used.
- To have the Hypercalls from El1 to El2, ARM SMCCC HVC is used.
- HvCallGetVpRegisters accesses registers beyond a0 to a3. For that SMCCC 1.2 is implemented.
- Code :
  - https://github.com/freebsd/freebsd-src/blob/main/sys/dev/psci/smccc_arm64.S
  - https://github.com/freebsd/freebsd-src/blob/main/sys/dev/psci/smccc.h



HYPER-V IDENTIFY AND LOAD

- Azure ARM64 hosts virtualizes the system counter and timer defined by the ARM64 architecture.
- Hyper-V synthetic timer counter initialization is not required here.
- hypercall page setup is moved from hyperv.c to x86 specific hyperv_x86.c, along with hyperv timer counter initialization.
- hyperv_et.c is also not required for ARM64, it is only built now for x86.
- Have used ACPI FADT to identify Hyper-V, which was done using CPUID in x86.
- Have used ARM SMCCC HVC to identify certain features of Hyper-V and to set the guest OS id.

HYPER-V VMBUS



VMBUS INTERRUPT HANDLING

- In x86 VMBus was using Free IDT vector for Hyper-V ISR.
  In ARM64 VMBus uses Interrupt mentioned in the _CRS of the HID VMBus.
- This resource is currently owned by vmbus_res as a direct child of ACPI.
- To access this resource from vmbus_res, we have used :
  `devclass_get_device(devclass_find("vmbus_res"),0)`
- Also introduced new attributes in vmbus_softc: ires, icookie and vector.

```
Name (_HID, "VMBus")  // _HID: Hardware ID
    Name (_UID, Zero)  // _UID: Unique ID
    ...
    Name (_CRS, ResourceTemplate ()  // _CRS: Current
Resource Settings
    {
      Interrupt (ResourceConsumer, Edge, ActiveHigh,
Exclusive, ,, )
      {
        0x00000012,
      }
```

CONTD.

- From the successful allocated ires resource, we are getting the irq number using rman_get_virtual(), which we are using then for synthetic interrupt controller setup.
  sc->vmbus_idtvec = irq_data->irq;
- These changes are in vmbus_aarch64.c and the lapic based IDT vector setup has been moved in vmbus_x86.c

CURRENT WORK

- Enabling vmbus_pcib for to use accelerated networking feature of Hyper-V in Azure.
  - Hyper-V does not emulate a full-fledged PCI bridge.
  - A cooperative PCI bridge driver is needed on FreeBSD.
    - Handle PCI configuration space accessing.
    - Setup BARs for SR-IOV/passed-through devices.
    - Remap MSI/MSI-X data and address.

- Following this work, we will have SR-IOV, NVME enabled for FreeBSD on ARM64 Hyper-V.

THANK YOU AND Q&A