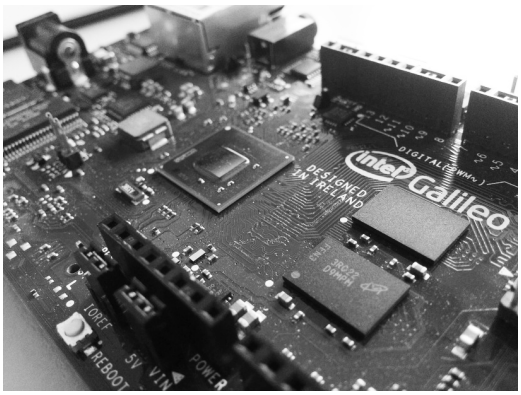
1 Übungsaufgabe 2: Konfigurierbares StuBSmI

In dieser Aufgabe soll ein StuBSmI-Port für das Intel Galileo-Board[1] entwickelt und mit Hilfe von `Kconfig` als Software-Produktlinie dargestellt und somit konfigurierbar in StuBSmI integriert werden.

1.1 Galileo-Board



Zur Entwicklung befinden sich in der *Manlobbi* (Raum 0.057-113) drei Exemplare dieses Boards an den Rechnern `faui49man{1,4,6}`. Die entsprechenden Accounts für die Manlobbi-Laborrechner erstellen wir euch während der Rechnerübungen.
Da das Board weder eine eigene grafische Ausgabe noch einen PS/2-Anschluss besitzt, wird die Ein- und Ausgabe über eine serielle Verbindung realisiert. Für die Bedienung des Bootmenüs – sowie später auch die Ein- und Ausgabe in StuBSmI – wird daher der jeweilige mit dem Board verbundene Rechner als ,,serielle Konsole" verwendet, indem man sich beispielsweise mit `screen /dev/ttyS0 115200` verbindet.
Im Unterschied zum x86-Testrechner können die Galileo-Boards nicht direkt per Netzwerk sondern nur von einer SD-Karte booten. Deswegen muss zusätzlich zum Aufruf von `make netboot` zunächst der Bootmenüeintrag `yocto linux: get images, reboot` gestartet werden, welcher die aktuellen Images auf die SD-Karte kopiert und das Board neu startet. Erst im danach erscheinenden GRUB-Menü kann das eigene System ausgewählt und gestartet werden.
Auf dem Galileo-Board kann der serielle Anschluss über einen *Universal Asynchronous Receiver Transmitter* (UART) gesteuert werden, dessen Register ab der physikalischen Adresse `0x9000b000` als *memory mapped registers* zugreifbar sind. Das Datasheet[2] der Galileo-Hardware beschreibt in Kapitel 18.6 die konkreten Register, welche am besten wie folgt konfiguriert werden:

- 115200 Baud
- 8 Bit Character, ein Stop-Bit, kein Parity-Bit
- FIFOs an + IRQ sobald ein Zeichen im Empfangs-FIFO ist
- alles was ,,modem control" betrifft auf 0 setzen

Die Interrupt-Leitung des UART liegt an Port 17 des I/O-APIC und löst mit einem *active low* Pegel (⇒ *trigger mode: level*) aus. Diese Information darf statisch zur Konfiguration verwendet werden und muss nicht zur Laufzeit via ACPI ermittelt werden.
Die serielle Ausgabe ist zeichenstromorientiert und unterscheidet sich damit grundsätzlich vom wahlfreien Zugriff auf den CGA-Bildschirm. Um die Flexibilität der `CGA_Screen`-Klasse zu bewahren, können jedoch ANSI-Kontrollsequenzen[3] dafür genutzt werden, den Cursor vor einer Zeichenausgabe frei zu positionieren.
Da der serielle Anschluss relativ träge ist, ist es wahrscheinlich, dass zum Zeitpunkt einer Unterbrechung bereits mehrere Zeichen im Eingabepuffer des UART auf Abholung warten. Es müssen alle Zeichen ausgelesen werden, da sonst für folgende Eingaben keine weitere Unterbrechung generiert wird. Um diese Aufgabe zu vereinfachen, kann die in Vorgabe dieser Übungsaufgabe[4] enthaltene ,,Bounded Buffer"-Implementierung verwendet werden.

1.2 Kconfig

Die unterschiedlichen Verhaltensweisen von StuBSmI für den Betrieb auf dem x86-PC und dem Galileo-Board StuBSmI Parameter sollen nun in ein Merkmalmodell (*feature model*) überführt werden. Die Umsetzung erfolgt in der Sprache des Linux-Konfigurationswerkzeugs `Kconfig`.
Die Idee dabei ist, dass man auch als Nicht-Domänenexperte, also ohne tiefere Kenntnis der StuBSmI-Interna eine spezifische Lösung für das eigene Szenario generieren kann. Für den Linux-Kern wurde dazu ein spezielles Werkzeug, `Kconfig`, entwickelt, welches es den Benutzern erlaubt, den Kern zu konfigurieren. `Kconfig` stellt sicher, dass die vom Benutzer getroffene Auswahl im Variantenmodell gültig ist und beeinflusst den Build-Prozess entsprechend, so dass die auswählte Variante erzeugt wird.
Eine aus dem Linux-Kern herausgelöste und teilweise modifizierte Version dieser Werkzeugkette ist Teil der Vorgabe für diese Übungsaufgabe und soll für die Umsetzung der Konfigurierbarkeit genutzt werden. Um im

[1] `http://www.intel.com/content/www/us/en/do-it-yourself/galileo-maker-quark-board.html`
[2] /proj/i4bs/doc/329676_QuarkDatasheet.pdf
[3] `https://en.wikipedia.org/wiki/ANSI_escape_code`
[4] /proj/i4bs/vorgaben/aufgabe-kss2.tar.gz

Konfigurationsmenü nicht nur auf einen einsamen „Galileo-Support ja/nein"-Schalter beschränkt zu sein, sollten noch weitere Merkmale und Parameter konfigurierbar gemacht werden, wie beispielsweise die Größen der Stacks, oder die Benutzung von Copy-On-Write (ab der dritten BST-Aufgabe), oder beliebige weitere Eigenschaften.

Hinweise:

- Das UART-Status-Register scheint nicht zu tun was es soll und enthält immer 0.
- Auch die Speicher-Seite des UART muss in der MMU-Konfiguration explizit abgebildet werden.
- QEMU hat auch einen UART (wenn auch an IO-Port `0x3F8` und IO-APIC Port 2), dieser lässt sich genauso konfigurieren und über "`-serial stdio`" wird die serielle Ein- bzw. Ausgabe auf `stdin` und `stdout` umgeleitet.
- Es treten sporadisch Interrupts mit der Vektornummer 39 auf; vermutlich durch aktive PCI-Geräte. Diese Interrupts dürfen ignoriert werden.
- Im Adressbereich des CGA-Gerätes befindet sich auf dem Galileo-Board kein benutztbarer Speicher, deswegen sollen dorthin keine weiteren Schreiboperationen erfolgen. Ebenso ist das Schreiben auf IO-Ports für den CGA-Cursor nicht mehr sinnvoll.

Abgabe: am 25.06.2014