

Høynivå Embedded Operativsystemer (Inventas) - inventas.no

Tradisjonelt sett er innebygde systemer enten basert på bare-metal/RTOS som kjører på små mikrokontrollere med begrensede ressurser, eller de drives av embedded Linux på kraftigere System-on-Modules (SoMs). Det er imidlertid en betydelig avstand mellom disse to tilnærmingene: det som er enkelt å implementere på bare-metal kan ofte være utfordrende i Linux. Linux ble ikke opprinnelig utviklet for innebygde systemer, og det å tilpasse det til spesialtilpasset maskinvare kan være en tidkrevende og kompleks prosess. Derfor er det behov for et høynivå embedded operativsystem som kan fylle dette gapet.

Redox er et operativsystem skrevet i Rust, og selv om det for tiden er rettet mot å være et all-purpose OS, har det egenskaper som potensielt gjør det svært egnet for innebygde systemer. Redox er basert på en mikrokjerne hvor alle drivere kjører i brukermodus, i motsetning til Linux, som er monolitisk med drivere i kjernerommet. Dette betyr at drivere i Redox er vanlige programmer, noe som forenkler driverutvikling sammenlignet med Linux. Hvis en brukermodusprosess i Redox for eksempel prøver å lese fra ugyldig minne, vil det resultere i en segmenteringsfeil uten å påvirke resten av systemet, mens i Linux kan en slik feil føre til at hele kjernen kræsjer og krever en omstart av systemet.

For øyeblikket støtter Redox kun x86, x86_64 og aarch64, men målet er å få det til å kjøre på ARMv7-A-arkitekturen.

Målsetting

Målet med denne oppgaven er å kjøre embedded Redox på en 32-bits Cortex-A-basert SoM, som for eksempel STM32MP1 eller NXP i.MX. Prosjektet har som ambisjon å få et fungerende system opp og kjøre på et spesifikt utviklingskit, med drivere for alle nødvendige periferienheter, inkludert GPIO, I2C, SPI, UART, USB og Ethernet. Systemet skal kunne boote opp i et shell som tilbyr effektive kommandolinjeverktøy for utforskning og manipulering av disse periferiene, som for eksempel å lese og skrive til I2C-enheter.

Forskningsmål

En viktig del av denne oppgaven er å undersøke om et RTOS basert på Redox-kjernen gir et tryggere system, en bedre utvikleropplevelse og mer deterministisk sanntidsplanlegging sammenlignet med embedded Linux. Dette vil innebære en vurdering av systemets pålitelighet, hvor enkelt det er å utvikle og vedlikeholde drivere og applikasjoner, samt hvor godt systemet håndterer tidskritiske operasjoner.

Forskningsmålene inkluderer:

- **Ytelsesevaluering:** Utføre benchmark-tester for å evaluere ytelsen og sanntidsegenskapene til det Redox-baserte systemet sammenlignet med embedded Linux. Dette inkluderer måling av latens, gjennomstrømning og kontekstbyttetider.

- **Sammenligning av utvikleropplevelse:** Analysere utviklingsprosessene i begge miljøene, inkludert driverutvikling og systemkonfigurasjon, for å avgjøre hvilket operativsystem som tilbyr en bedre utvikleropplevelse.
- **Analyse av systempålitelighet og sikkerhet:** Undersøke hvordan brukermodusdrivere i Redox påvirker systemstabilitet og sikkerhet sammenlignet med kjerneromsdrivere i Linux. Dette innebærer å utføre stresstester og sikkerhetsanalyser.
- **Ressursutnyttelsesanalyse:** Vurdere effektiviteten til begge systemene når det gjelder minneforbruk og prosesseringsbelastning, spesielt i ressursbegrensede miljøer.
- **Implementering av prioritetsbasert planleggingsalgoritme:** Utvikle og integrere en planlegger som støtter sanntidsprioriteter i Redox, og evaluere dens effektivitet i håndtering av tidskritiske oppgaver.

Oppgaver

1. Legge til støtte for ARMv7-A i Redox-kjernen:
 - Modifisere Redox-kjernen for å støtte ARMv7-A-arkitekturen, og sikre kompatibilitet med den valgte SoM-maskinvaren.
2. Utvikle en UART-driver for SoM-en:
 - Implementere en UART-driver i Redox for seriell kommunikasjon, og validere driverens funksjonalitet med testapplikasjoner.
3. Boot systemet med fungerende terminal over UART:
 - Konfigurere bootloader og kjerne for å initialisere UART-konsollen, og oppnå et fungerende terminalgrensesnitt for brukerinteraksjon.
4. Implementere drivere for GPIO, I2C, SPI, UART, USB og Ethernet:
 - Utvikle og teste drivere for essensielle periferienheter, og sikre at driverne opererer pålitelig innenfor brukermodusmiljøet.
5. Utvikle kommandolinjeverktøy for utforskning av periferiene:
 - Lage verktøy for å interagere med og manipulere periferienheter, og lage dokumentasjon.
6. Implementere en prioritetsbasert planleggingsalgoritme i Redox:
 - Designe og integrere en planlegger som støtter sanntidsprioriteter, inkludert å teste planleggerens effektivitet i håndtering av tidskritiske oppgaver.
7. Gjennomføre benchmark-tester:
 - Måle systemets ytelsesmetrikker som latens, gjennomstrømning og kontekstbyttetider, og evaluere sanntidsrespons under ulike belastningsforhold.
8. Sammenligne utviklingsprosesser mellom Redox og embedded Linux:
 - Dokumentere trinnene som kreves for driverutvikling og systemkonfigurasjon i begge miljøene, og identifisere utfordringer og fordeler.

9. Analysere systempålitelighet og sikkerhet:

- *Utføre stresstester for å vurdere systemstabilitet, og evaluere hvordan brukermodusdrivere i Redox påvirker total systemsikkerhet sammenlignet med kjerneromsdrivere i Linux.*

10. Ressursutnyttelsesanalyse:

- *Sammenligne minne- og CPU-bruk mellom Redox og embedded Linux, og vurdere effektiviteten til begge systemene i ressursbegrensede miljøer.*
- *Ressursbruk under bygging (disk, RAM, CPU-tid), størrelse på disk.*

11. Dokumentere funn og gi anbefalinger:

- *Oppsummere resultatene av ytelsesevalueringer og sammenlignende analyser.*

Intern kontakt: yngveha@ifi.uio.no

Inventas kontakt: marius.elvegard@inventas.no