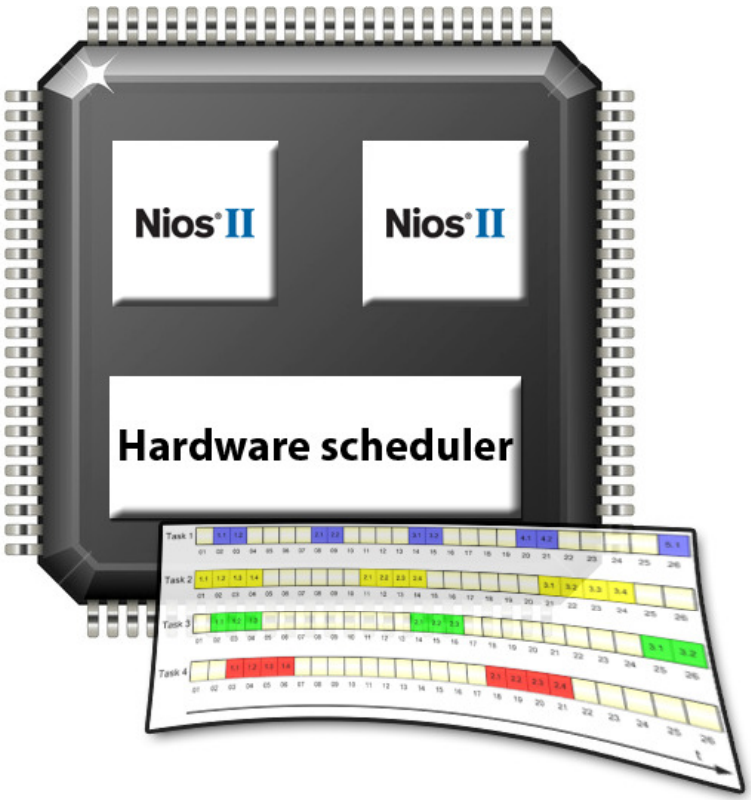
ALMA MATER STUDIORUM – UNIVERSITY OF BOLOGNA
DEIS - DEPARTMENT OF ELECTRONICS, COMPUTER ENGINEERING AND SYSTEMS



Hard real-time meets embedded multicore NIOS SoPCs

Primiano Tucci

Università di Bologna

DEIS – Dipartimento di Elettronica, Informatica e Sistemistica

Il paradigma System on (Programmable) Chip come fattore abilitativo per le tecnologie Real-Time embedded

- Consolidamento della infrastruttura hardware, integrando:
  - Piattaforme computazionali general-purpose (NIOS-II soft-processor)
  - Hardware custom special-purpose
  - Acceleratori / IP Core di terze parti (FFT, AES, MAC …)

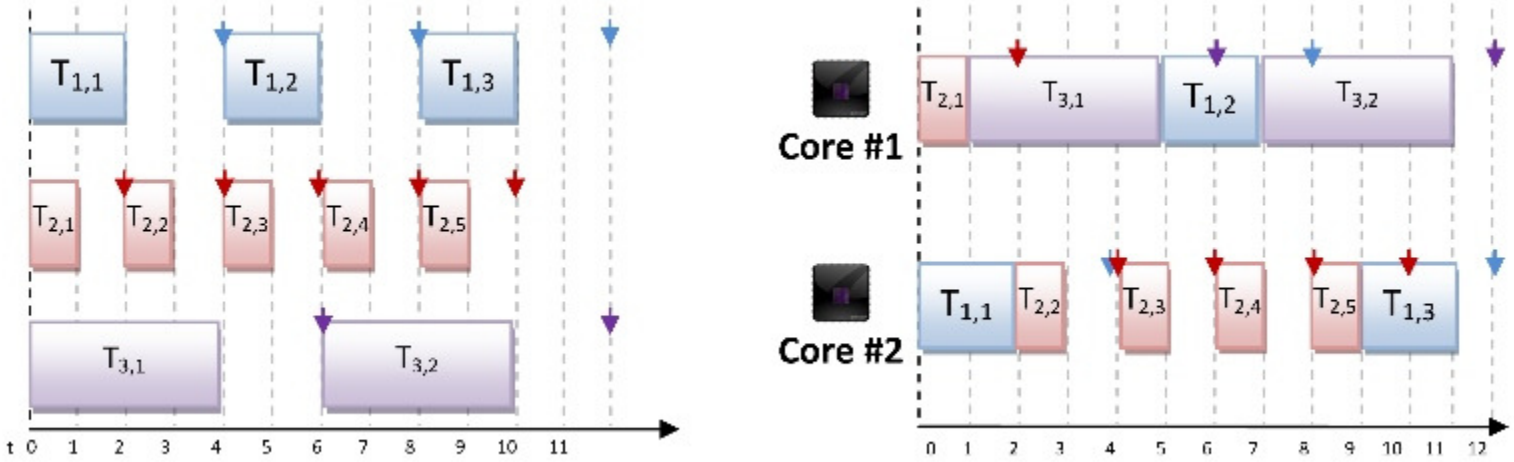
- Possibilità di realizzare configurazioni multi-processore

Nuove sfide sia per quanto riguarda sia il software che l'hardware.

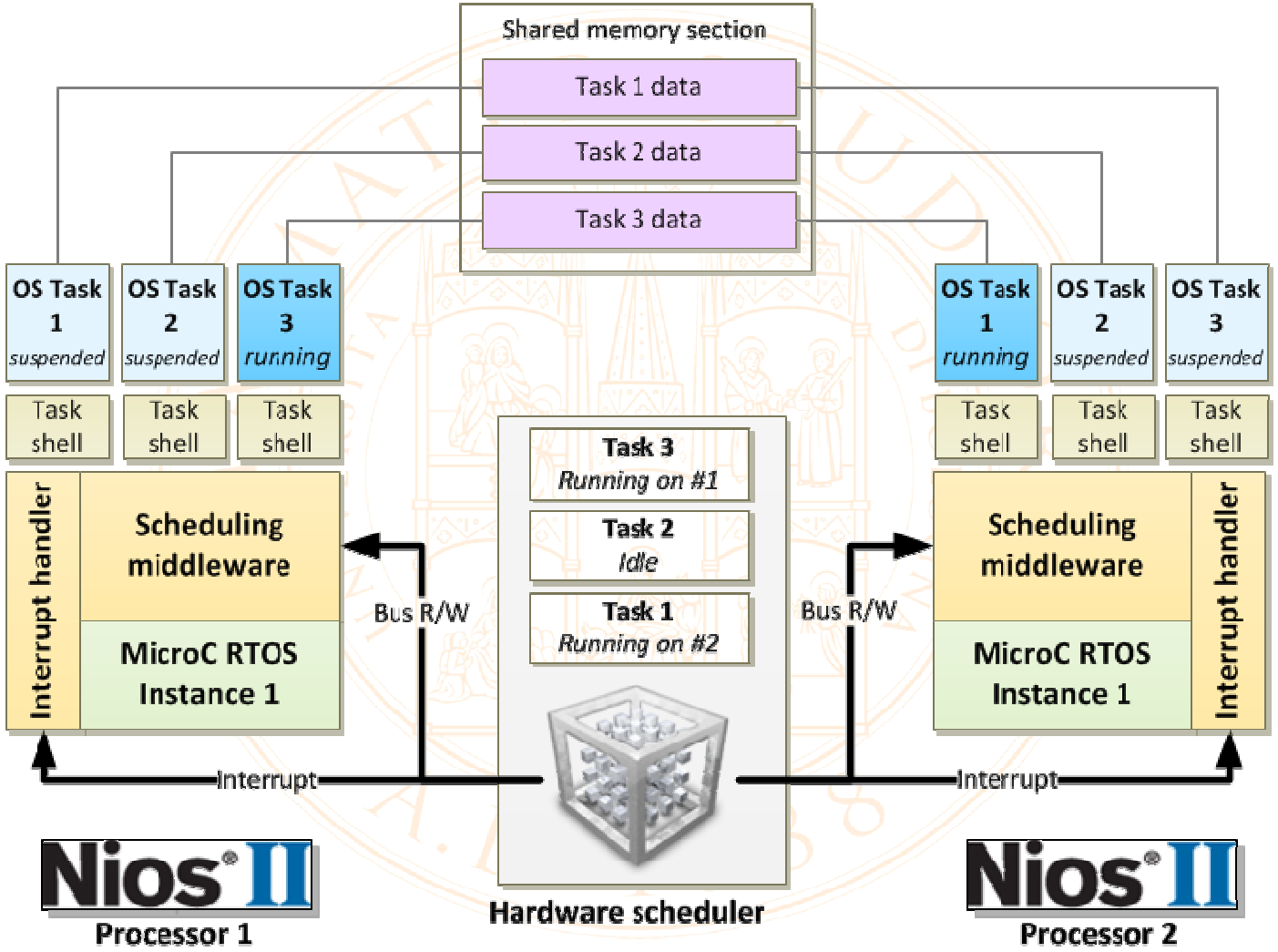
Obiettivo del progetto

Piattaforma real-time multi-processore

- Definizione della architettura hardware
- Definizione della architettura software
- Design dei componenti necessari alla integrazione HW/SW



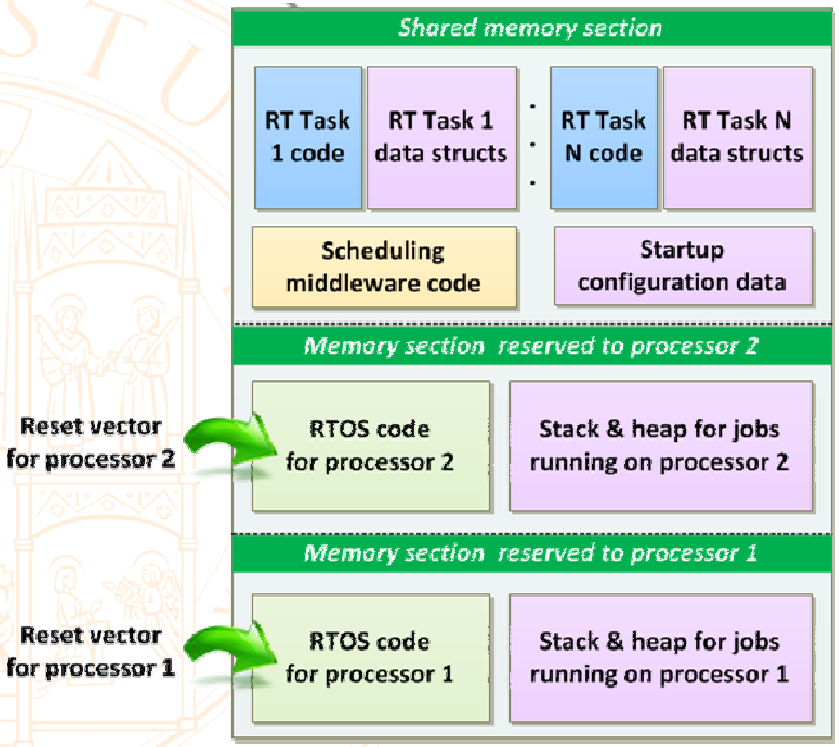
Architettura software



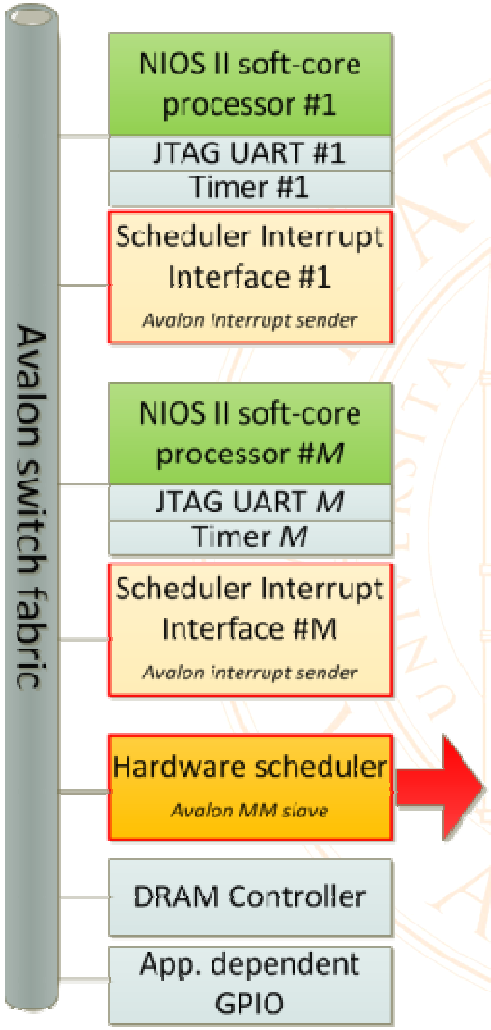
Layout software

Problematiche

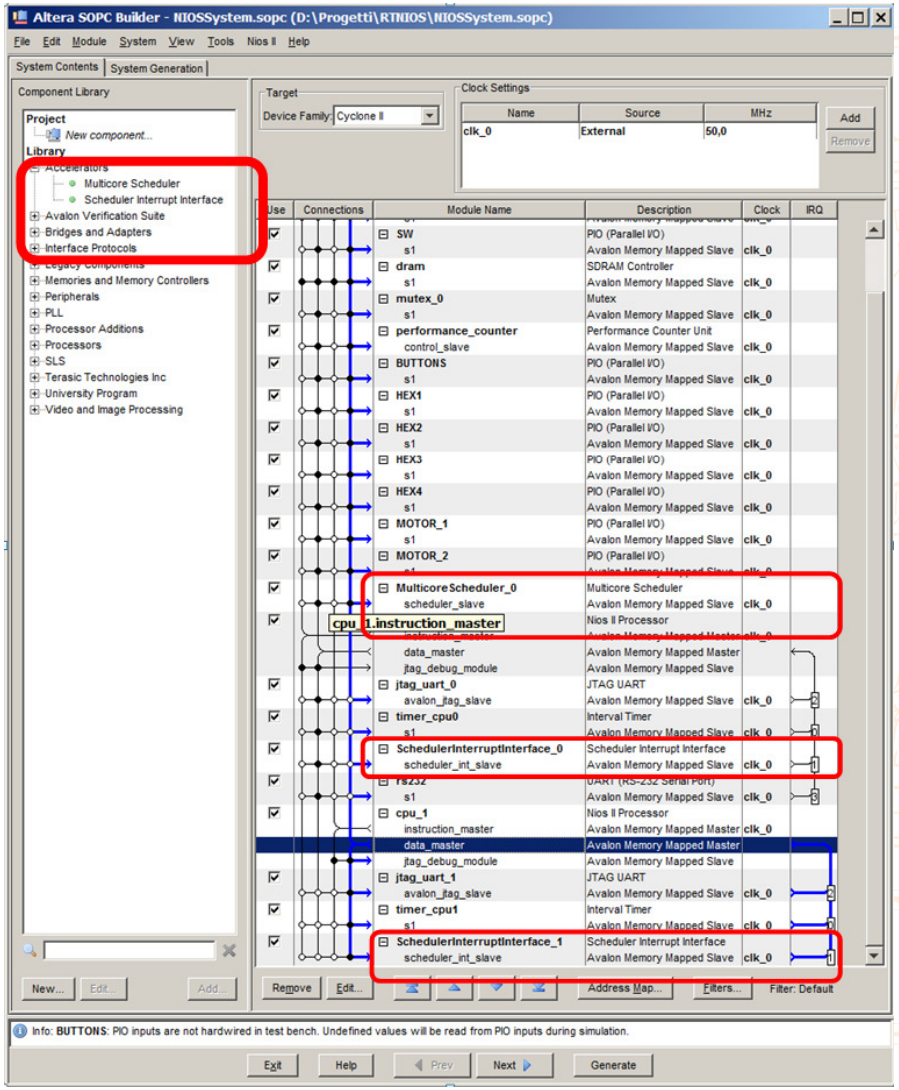
- Inapplicabilità dell'approccio SMP (coerenza cache, meccanismi IPC, maggiore complessità software)
- Approccio AMP, istanze indipendenti del RTOS su ogni processore.
- Coordinare dinamicamente l'avanzamento dei task tra i vari RTOS, garantendo il rispetto dei vincoli temporali
- Intervenire in modo *portabile* senza modificare il sistema operativo ma sfruttando le primitive da esso messe a disposizione.



Architettura hardware

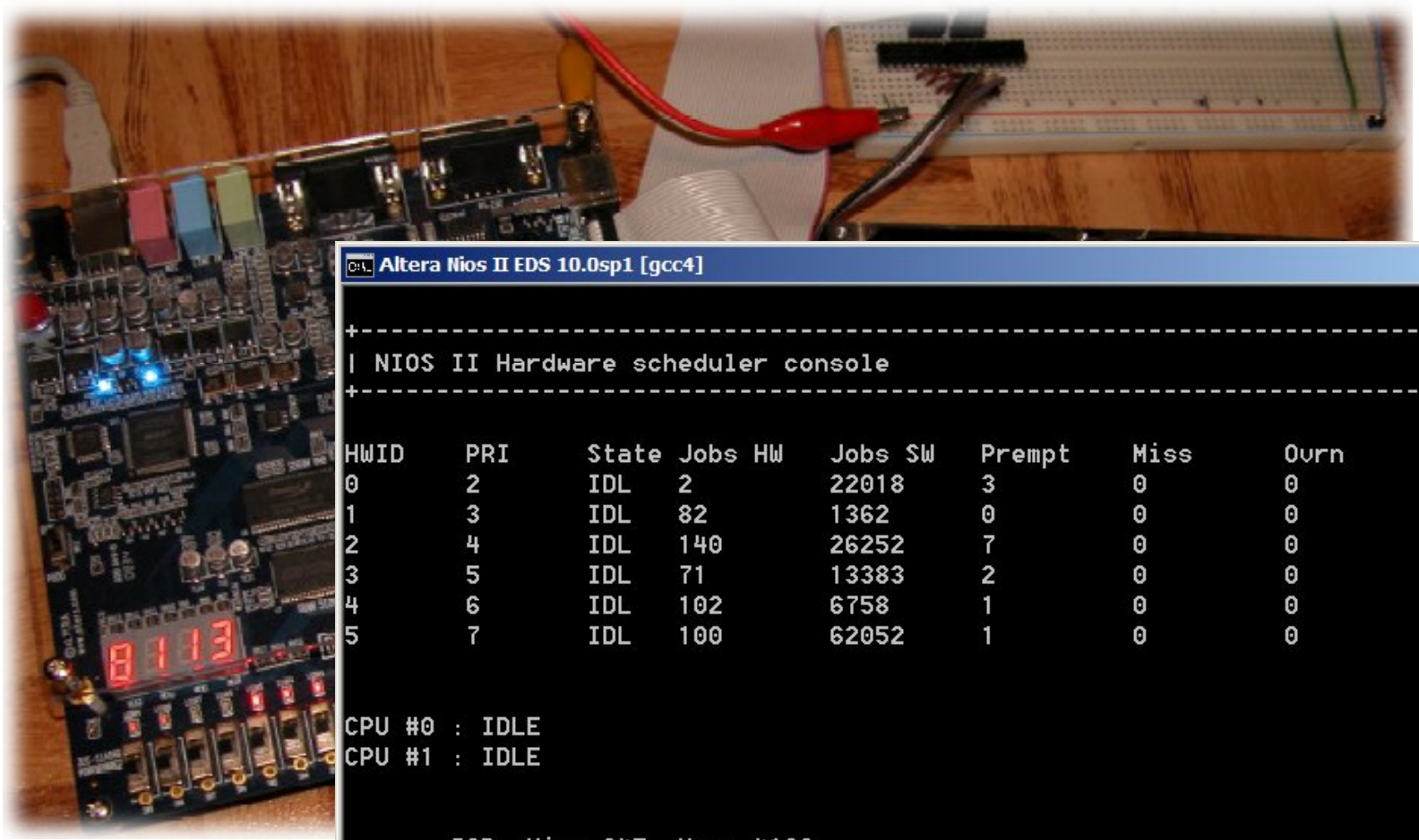


Integrazione nella piattaforma SoPC




```
void SharedSystemStartup()
{
  ApiResult res;
  printf("Welcome to HWScheulder on CPU %d\n",
get_cpuid());

  hwsched_uc_init();

  res = hwsched_uc_create_periodic_task(
    20,                 /* Task period          */
    20,                 /* Task deadline        */
    motor_control,      /* Job entry point      */
    &motor_conrol_data /* Job data structures */
    );

  printf("Motor control task created, res = %d\n",res);

  /* Repeat the same for each real-time task */

  printf("Entering real-time mode\n");
  res = hwsched_uc_start();
  if (res != S_OK)
  {
    printf("Error during scheduler initialization\n");
    for(;;) {}
  }
}
```

Implementazione



Risultati

- Offload delle funzioni di scheduling
  La politica di scheduling è completamente gestita in hardware, riducendo al minimo l'overhead computazionale richiesto ai soft-processor.

Overhead interrupt service routine		Overhead task shell	
Min	**Max**	**Min**	**Max**
245 T CLK (4.9 us)	4189 T CLK (83.78 us)	245 T CLK (4.9 us)	1249 T CLK (24.98 us)

- Basso jitter nel rilascio di processi periodi (10 ns @ 50 MHz)
- Adattabilità dell'hardware in base alle necessità

Configuration	Total comb. functions	Total logic registers	f_{MAX}
2 cores, 8 Tasks, 8 bit deadline and period counters, 8 bit stats counters	1246 (7%)	598 (3%)	59.8 MHz
4 cores, 16 Tasks, 8 bit deadline and period counters, 8 bit stats counters	2363 (13%)	1114 (6%)	51.86 MHz
8 cores, 32 Tasks, 16 bit deadline and period counters, 16 bit stats counters	7167 (38%)	3738 (20%)	39.08 MHz

Synthesis on a Cyclone II EP2C20F484C7 using the Quartus II EDA (Opt.: balanced) and the TimeQuest timing analyzer (slow-model)

Grazie per l'attenzione