

```
entity test_shift is
  generic ( width : integer := 17 );
  port ( clk : in std_ulogic;
        reset : in std_ulogic;
        load : in std_ulogic;
        en : in std_ulogic;
        outp : out std_ulogic );
end test_shift;
```

# Mikroprocesorska elektronika

## Predavanje I

```
shifter : process (en, reset)
begin
  if (reset = '0') then
    shift_reg <= (others => '0');
  elsif rising_edge ( clk ) then
    if (load = '1') then
      shift_reg <= unsigned (inp);
    elsif ( en = '1' ) then
```

# Sadržaj predavanja

- Uvod u embeded sisteme
- Struktura embeded sistema
- Klasifikacija embeded sistema

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```



```
entity test_shift is
  generic ( width : integer := 17 );
  port ( clk : in std_ulogic;
        reset : in std_ulogic;
        load : in std_ulogic;
        en : in std_ulogic;
        outp : out std_ulogic );
end test_shift;
```

# Uvod u embeded sisteme

```
shifter : process (en, reset)
begin
  if (reset = '0') then
    shift_reg <= (others => '0');
  elsif rising_edge ( clk ) then
    if (load = '1') then
      shift_reg <= unsigned (inp);
    elsif ( en = '1' ) then
```

# Definicija embeded sistema

- Embeded sistem se uopšteno može opisati kao uređaj koji sadrži **tesno povezane hardverske i softverske komponente** projektovane sa ciljem ostvarivanja jedne ili više **unapred definisanih funkcija**
- Najčešće čini deo većeg sistema, nevidljiv je za krajnjeg korisnika
- Nije namenjen da može da se programira i modifikuje od strane korisnika
- Očekuje se da će raditi sa minimalnom ili bez ikakve interakcije sa čovekom
- Dve karakteristike su prisutne u većini embeded sistema:
  - **Reaktivno ponašanje**
  - **Rad pod vrlo velikim ograničenjima**

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Reaktivno ponašanje

- Većina embeded sistema **interaguje direktno** sa svojim okruženjem i procesima
- Odluke je potrebno donositi tokom samom rada, na osnovu **trenutnih vrednosti** ulaza u sistem
- Ovo zahteva da embeded sistemi budu **reaktivni**, reagujući u **realnom vremenu na promene ulaza**, na taj način obezbeđujući korektnu funkcionalnost

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Rad pod velikim ograničenjima

- Embedded sistemi rade u okruženjima koja se karakterišu **velikim ograničenjima**
- Raspoloživa **količina memorije, procesorske moći i energije** su tipično ograničeni
- **Zahtevi proizvodnje**, povezani sa brojem fabrikovanih uređaja, postavljaju veliko ograničenje u pogledu cene fabrikacije

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```



```
entity test_shift is
    generic ( width : integer := 17 );
    port ( clk : in std_ulogic;
          reset : in std_ulogic;
          load : in std_ulogic;
          en : in std_ulogic;
          outp : out std_ulogic );
end test_shift;
```

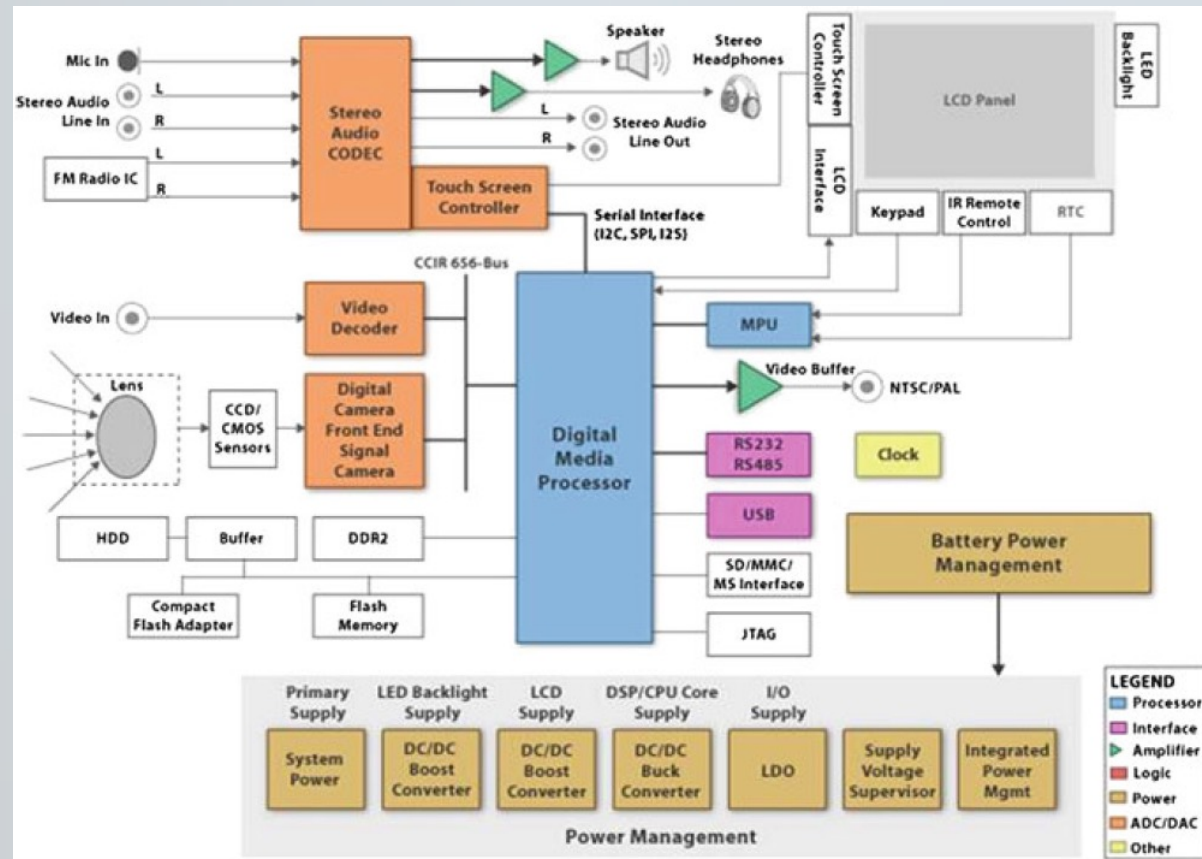
# Struktura embeded sistema

```
shifter : process (en, reset)
begin
    if ( reset = '0' ) then
        shift_reg <= (others => '0');
    elsif rising_edge ( clk ) then
        if (load = '1' ) then
            shift_reg <= unsigned (inp);
        elsif ( en = '1' ) then
```



# Struktura savremenog embeded sistema I

- Većina savremenih embeded sistema bazirana je na **mikroprocesorima**
- Usled složenosti aplikacija one se uglavnom realizuju pomoću **većeg broja interagujućih embeded sistema**
- Generički **multimedijalni uređaj**, prikazan na slici desno, sastoji se iz:
  - sistema za obradu audio signala
  - sistema za obradu video signala
  - sistema za smeštanje podataka
  - korisničkog interfejsa
  - sistema za napajanje
- Svaki od ovih sistem realizovan je **kao odvojeni embeded sistem** koji je integrisan u ciljnu aplikaciju





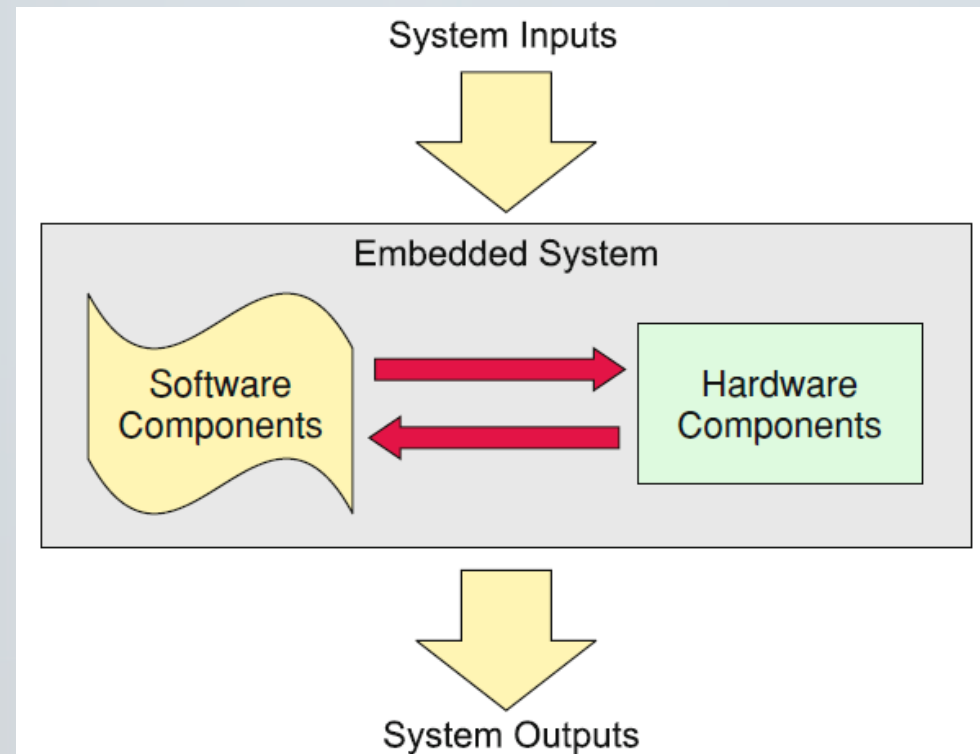
# Struktura savremenog embeded sistema II

- Prethodni slajd ilustrovao je koncept embeded sistema za vrlo specifičnu aplikaciju
- Međutim, sličan koncept sistema može se pronaći u gotovo svakom aspektu našeg svakodnevnog života:
  - **Multimedijalnim uređajima** - mobilni telefoni, digitalne kamere
  - **Kućnim aparatima** - mikrotalasna rerna, mašina za pranje veša i sudova, TV
  - **Transportnim uređajima** – kola, brodovi, vozovi, avioni
  - **Medicinskim uređajima i uređajima za održavanje života** – pejsmejker, rentgen
  - **Sigurnosnim sistemima** – ABS, airbag, električni nadzor
  - **Vojnim sistemima** – sistemi za navođenje projektila, radari, GPS sistemi
- Iako sveprisutni, embeded sistemi su gotovo neprimetni za krajnje korisnike obezbeđujući “inteligentan” rad mašina i uređaja oko nas

```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```

# Struktura savremenog embeded sistema III

- Bez obzira na funkciju koju izvodi, svaki embeded sistem se na najvišem nivou može podeliti na **dve, usko povezane**, komponente:
  - Skup **hardverskih komponenti**, koje tipično uključuju neku vrstu centralno procesirajućeg uređaja realizovanog u formi **mikroprocesora**
  - Skup **softverskih programa**, poznatih pod nazivom **firmware**, koji obezbeđuju željenu funkcionalnost hardvera
- Svaki embeded sistem interaguje sa svojim okruženjem preko:
  - **Ulaza u sistem**, koji predstavljaju **procesne promenljive** i **parametri** koji se najčešće prikupljaju preko različitih senzora i ulazno/izlaznih portova
  - **Izlaza iz sistema**, koji predstavljaju **kontrolne akcije** na odgovarajućim aktuatorima ili obrađene informacije za korisnike ili druge podsisteme u okviru ciljne aplikacije.



```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```

# Primer strukture embeded sistema I

- Analizirajmo strukturu embeded sistema ugrađenog u mikrotalasnu rernu
- Hardverska komponenta ovog sistema uključuju:
  - **magnetron** - generator mikrotalasa,
  - modul energetske elektronike – namenjen kontroli magnetrona
  - motor - za obrtanje ploče na koju se stavlja obrok
  - tastaturu – sa tasterima za izbor načina i trajanja pripremanja jela
  - sistemski displej – za prikazivanje statusnih i vremenskih informacija
  - zvučnik – za generisanje različitih audio signala
  - ugrađeni mikroprocesorski sistem – za koordinaciju rada čitavog sistema

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Primer strukture embeded sistema II

## ■ **Ulazi u sistem** uključuju:

- selekciju načina pripreme jela, vreme pripreme jela i jačinu grejanja – zadate preko tastature
- temperaturu jela
- statusne signale sa različitih unutrašnjih senzora i prekidača

## ■ **Izlazi sistema** uključuju:

- prestalo vreme pripreme jela i status mikrotalasne rerne – prikazane na sistemskom displeju
- trenutnu snagu magnetrona koja se prosleđuje sistemu za upravljanje
- komande da li je potrebno uključiti ili isključiti rotirajuću ploču
- signale ka zvučniku potrebne za generisanje različitih audio signala

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Primer strukture embeded sistema III

- **Softverska komponenta** predstavlja najapstraktniji deo embeded sistema, ali je podjednako važna za ostvarivanje ukupne funkcionalnosti sistema kao i hardverska komponenta
- Softverska komponenta uključuje **skup različitih programa** koji diktiraju sekvencu po kojoj rade hardverske komponente.
- Na primer, kada korisnik odabere neki od unapred programiranih načina pripreme jela, softver:
  - analizira pritisnute tastere na kontrolnom panelu i identifikuje koju je opciju odabrao korisnik
  - donosi odluku o potrebnoj jačini generisanih mikrotalasa i vremenu pripreme jela
  - započinje i prekida proces mikrotalasnog ozračivanja unutar rerne, uključuje i isključuje rotiranje ploče na koju je obrok postavljen
  - generiše potrebne audio signale kojima se korisnik obaveštava da je obrok pripremljen

```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```

# Primer strukture embeded sistema IV

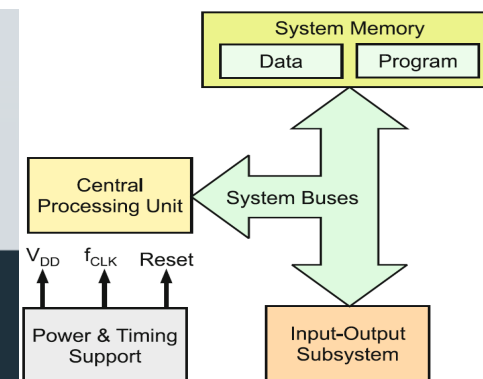
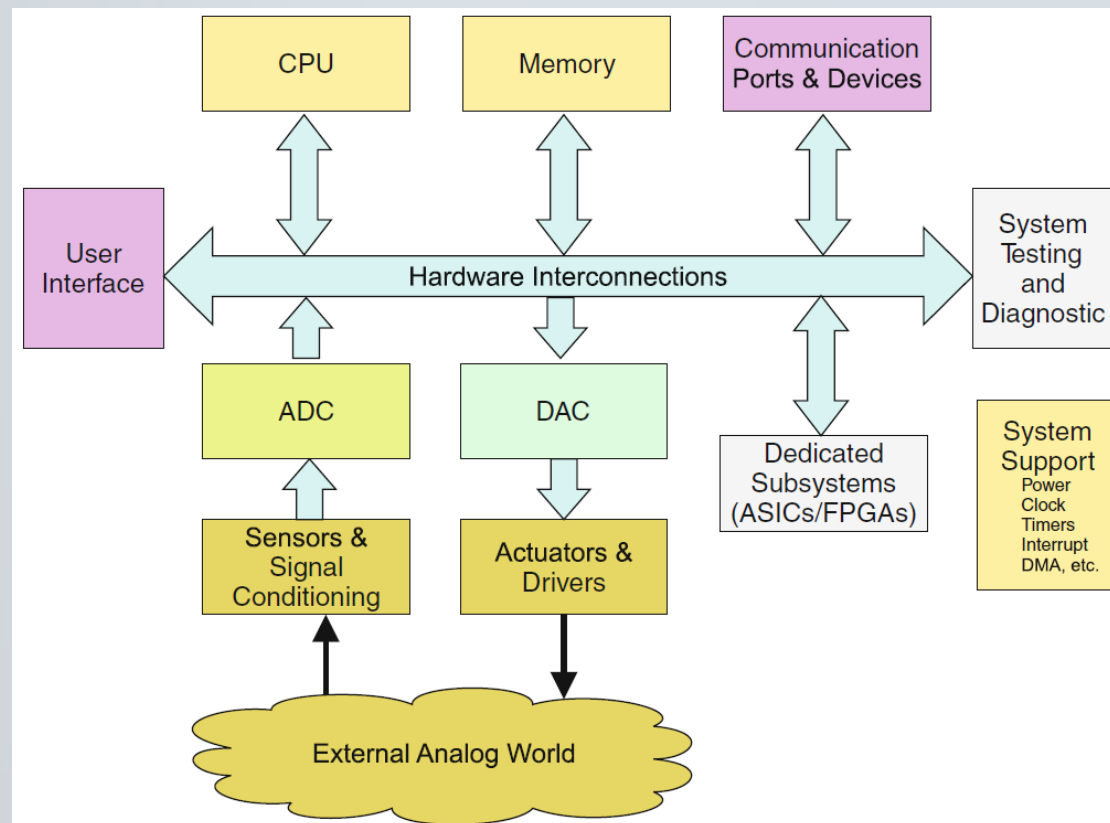
- U toku same pripreme jela, softver:
  - nadgleda temperaturu jela i, po potrebi, prilagođava jačinu i vreme trajanja generisanih mikrotalasnih signala
  - nadgleda ispravan rad unutrašnjih komponenti i u slučaju nepravilnog rada zaustavlja proces pripreme jela da bi se izbegle moguće katastrofalne posledice
- Iako je opis rada mikrotalasne rerne izveden na **sistemskom nivou**, tesna međuzavisnost između aplikacije, hardvera i softvera je očigledna.
- U nastavku će hardverska i softverska komponenta biti analizirane sa većim detaljima

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```



# Hardverska komponenta embeded sistema I

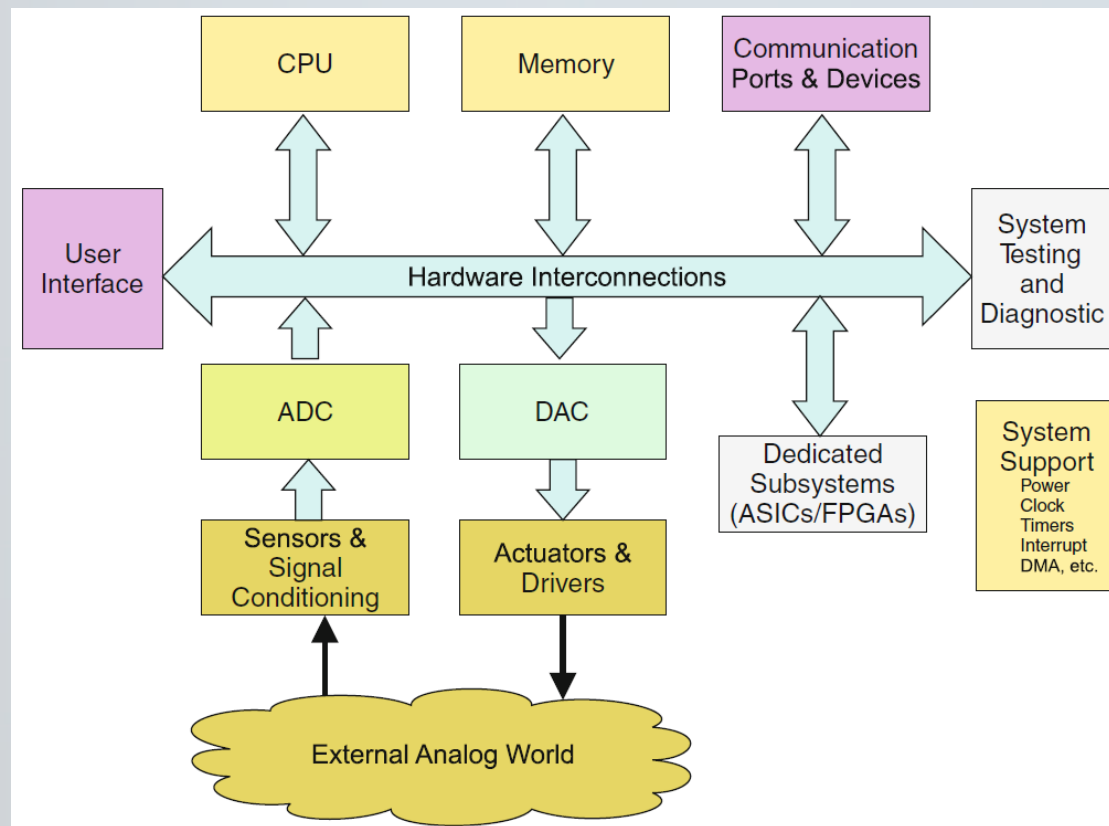
- Generalno, **hardverska komponenta** embeded sistema uključuje **sve elektronske komponente** neophodne da bi sistem mogao da ostvaruje funkciju za koju je projektovan.
- Imajući u vidu prethodnu definiciju, jasno je da će se hardverska struktura jednog embeded sistema značajno razlikovati od strukture drugog embeded sistema, ukoliko se razlikuju aplikacije za koje su ovi sistemi projektovani
- Ipak, **tri ključne hardverske komponente** svakog embeded sistema su:
  - CPU** - centralna procesirajuća jedinica
  - sistemska memorija**
  - ulazno/izlazni podsistem** - skup ulaznih i izlaznih portova preko kojih CPU komunicira sa svojim okruženjem





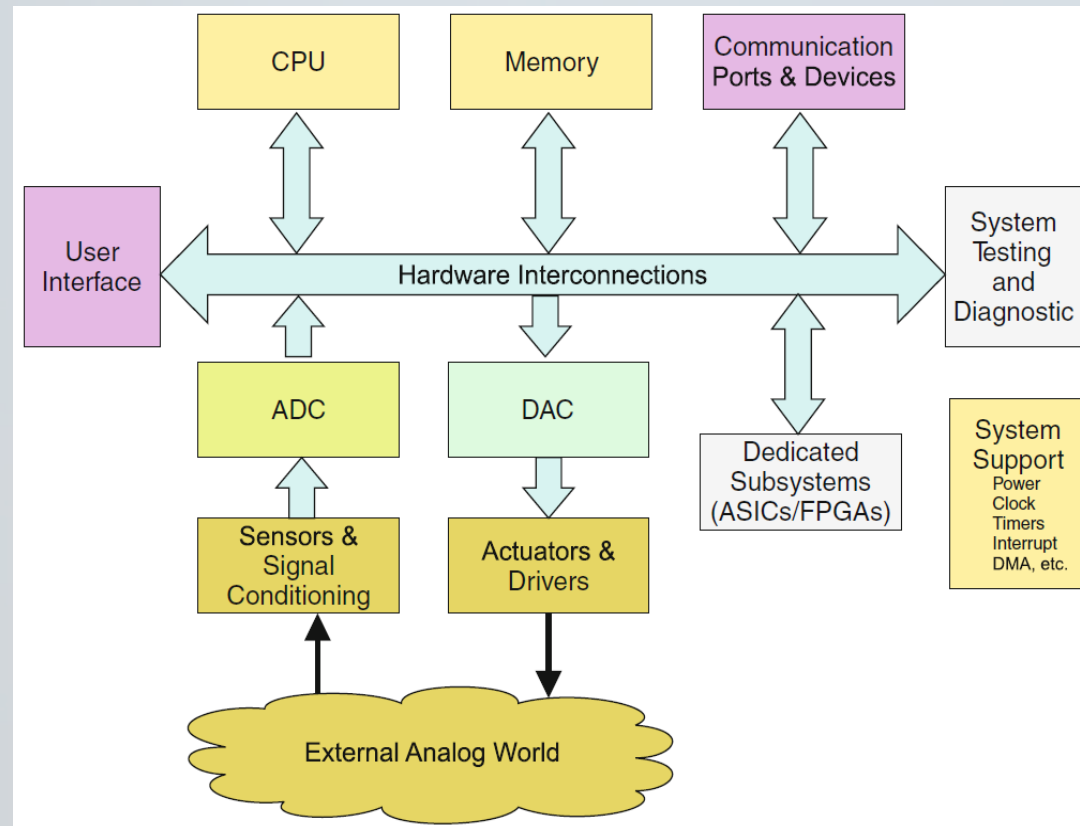
# Hardverska komponenta embeded sistema II

- **Centralni procesor (CPU)** izvršava instrukcije programa (softvera) sa ciljem obrade signala koji dolaze sa sistemskih ulaza i donošenja odluka koji upravljaju radom čitavog sistema
- **Sistemska memorija** skladišti programe i podatke neophodne da bi čitav sistem funkcionisao
- U većini embeded sistema postoji jasna podela između **programske memorije** i **memorije za podatke**
- **Programska memorija** služi za smeštanje programa koji se izvršava na CPU
- **Memorija podataka** čuva podatke koji se obrađuju u sistemu



# Hardverska komponenta embeded sistema III

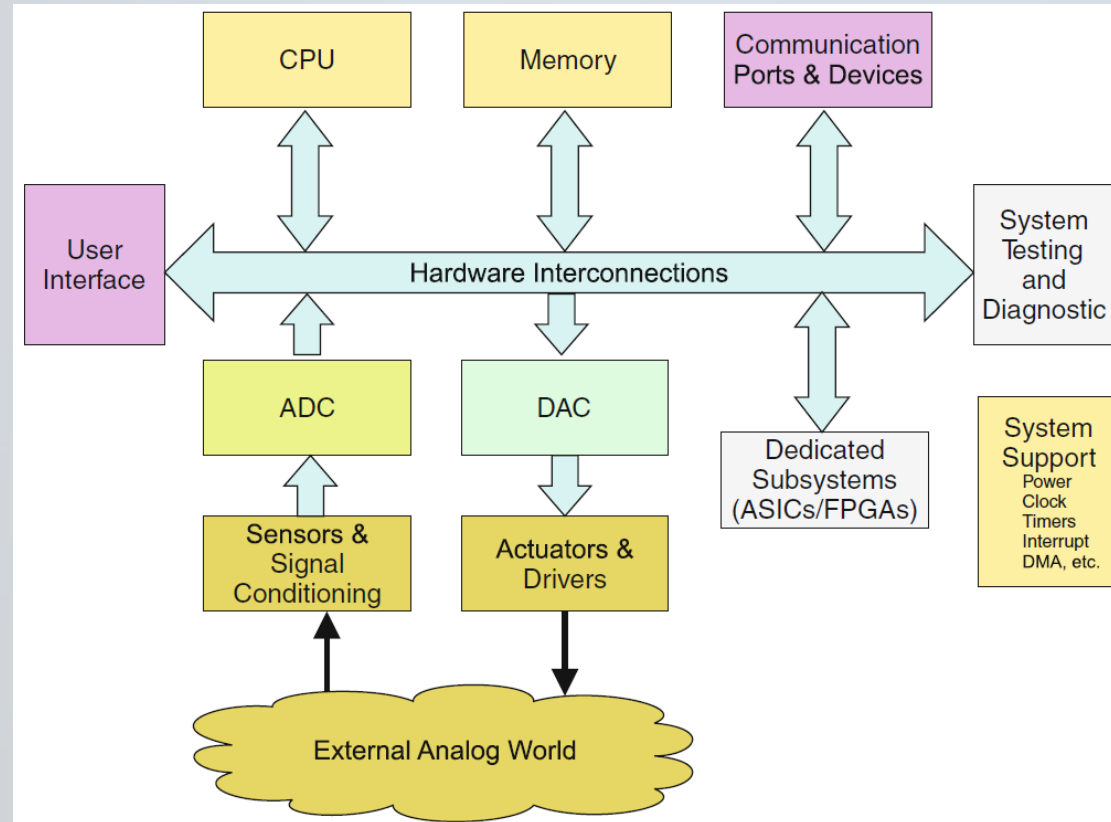
- **Ulazno/izlazni podsistem** omogućava prenos podataka između CPU i njegovog okruženja
- Pored navedene tri komponente, različit broj i vrsta **ulazno/izlaznih uređaja**, neophodnih za funkcionisanje sistema, može biti prisutan u zavisnosti od ciljne aplikacije:
  - **Komunikacioni portovi** za serijsku i/ili paralelnu razmenu podataka sa ostalim uređajima ili sistemima. USB portovi, printer portovi, bežični RF i infracrveni portovi su neki od mogućih primera.
  - **Korisnički interfejs** za interakciju sa ljudima. Tastature, prekidači, zujalice i mikrofoni, svetla, numerički, alfanumerički i grafički displeji su neki od mogućih primera.



```
shift_reg <= unsigned (inp);  
elsif (en = '1') then
```

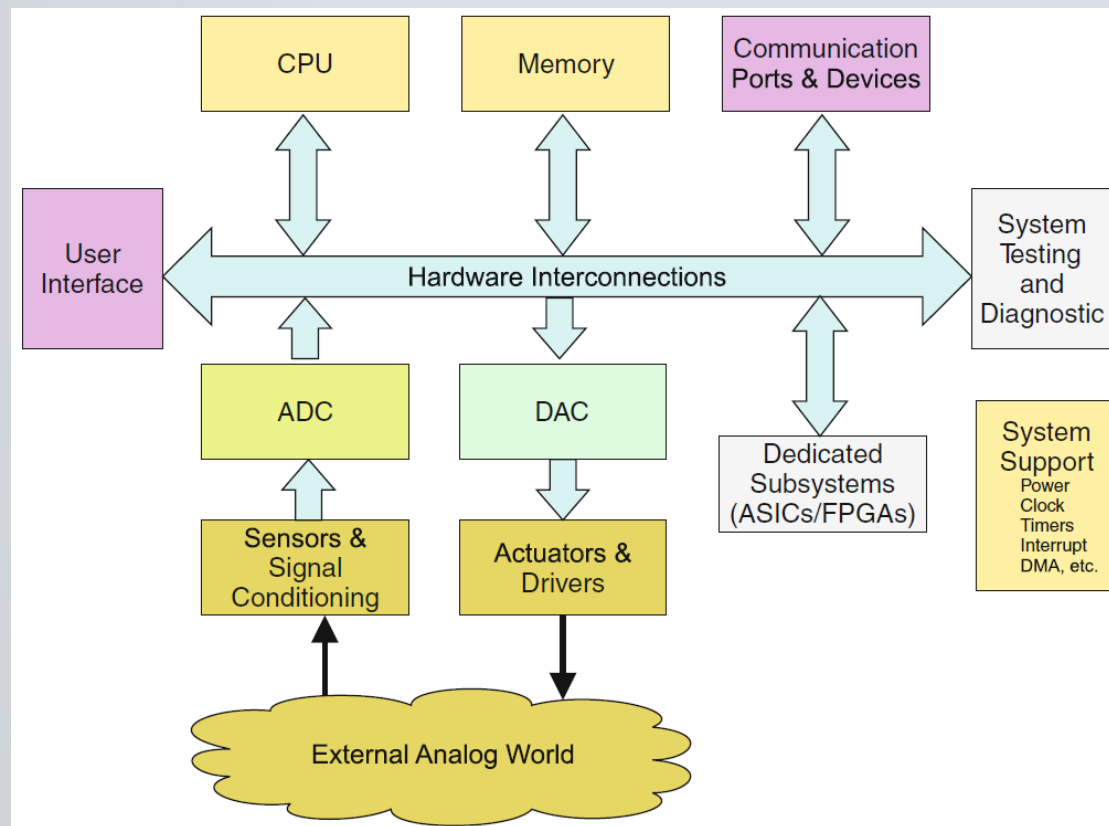
# Hardverska komponenta embeded sistema IV

- **Senzori i elektromehanički aktuatori** za interakciju sa okruženjem sistema
  - **Senzori** obezbeđuju **ulazne električne signale** koji su u vezi za različitim fizičkim veličinama kao što su: temperatura, pritisak, pomeraj, ubrzanje, rotacija, itd
  - Motori, step motori, releji su neki od primera aktuatora koji prihvataju i **transformišu električne** signale generisane od strane embeded sistema
- Analogno-digitalne (ADC) i digitalno-analogne (DAC) konvertore, koji omogućavaju interakciju sa analognim senzorima i aktuatorima. Ukoliko je signal koji dolazi sa senzora analogan, ADC ga pretvara u odgovarajući digitalni format koji se može obraditi pomoću CPU. Slično, kada CPU ima potrebu da zada komandu nekom od analognih aktuatora, DAC je neophodan da bi se promenio format signala.
- **Dijagnostičke i redundantne komponente** za verifikaciju i obezbeđivanje robusnog i pouzdanog rada čitavog sistema



# Hardverska komponenta embeded sistema V

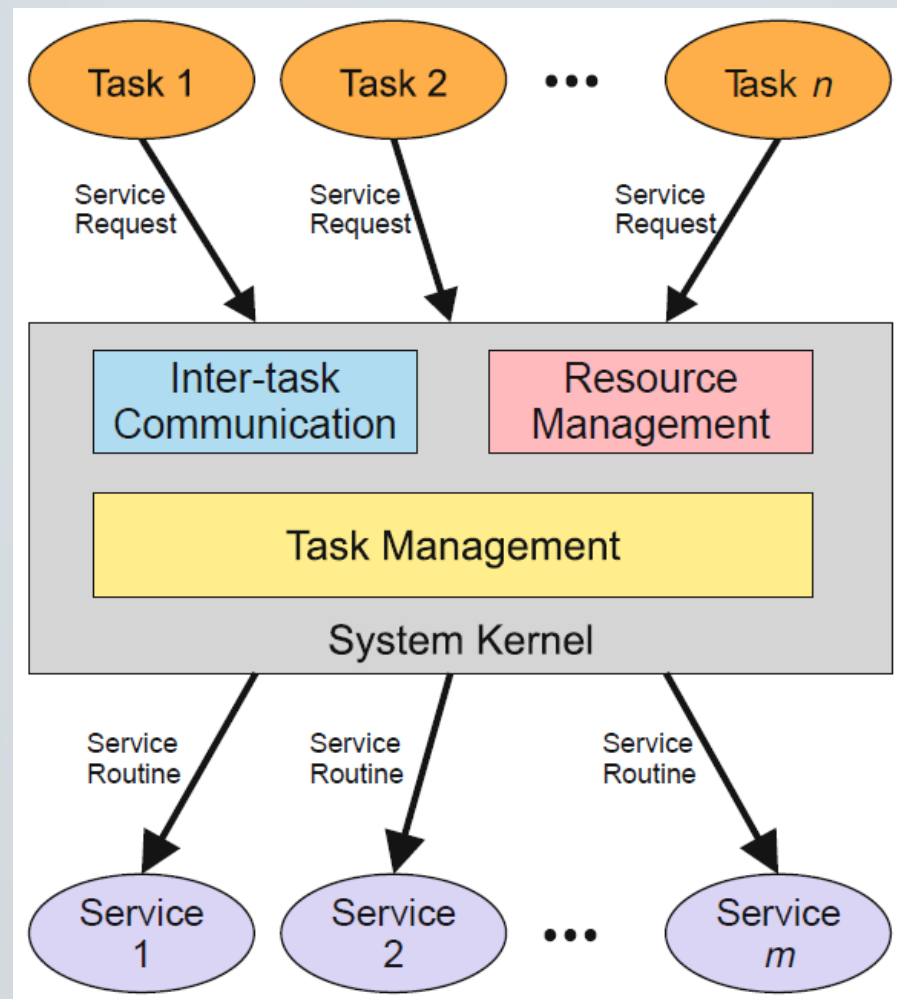
- Različite komponente koje obezbeđuju **esencijalne funkcije** koje omogućavaju nesmetan rad sistema. Ove komponente uključuju sisteme napajanja, generatore klok signala, tajmere, interapt kontrolere, DMA kontrolere, itd.
- Različite podsisteme koji realizuju **specifične funkcije sistema**. Ovi podsistemi mogu biti realizovani pomoću ASIC, FPGA ili drugih tipova namenskih komponenti, u zavisnosti od složenosti ciljne aplikacije.



```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Softverska komponenta embeded sistema I

- **Softverska komponenta** embeded sistema uključuje sve programe potrebne da se obezbedi zahtevana funkcionalnost hardvera
- Ovi programi poznati su pod nazivom **firmver** (firmware)
- **Sistemske programi** su obično organizovani oko nekog **operativnog sistema** i **aplikativnih rutina**
- Operativni sistem može biti mali i neformalan u slučaju malih aplikacija, ali kako aplikacije postaju složenije, operativni sistem će zahtevati više strukture i formalnosti

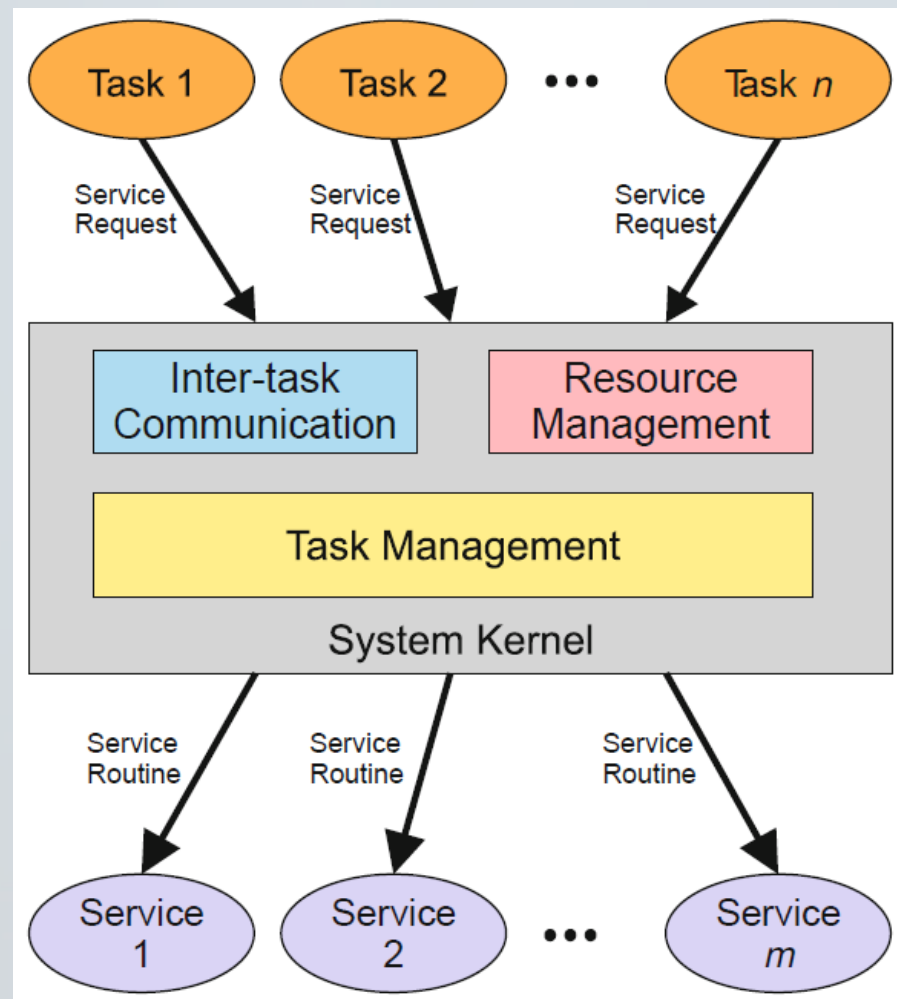


```
shift_reg <= unsigned (inp);  
elsif (en = '1') then
```



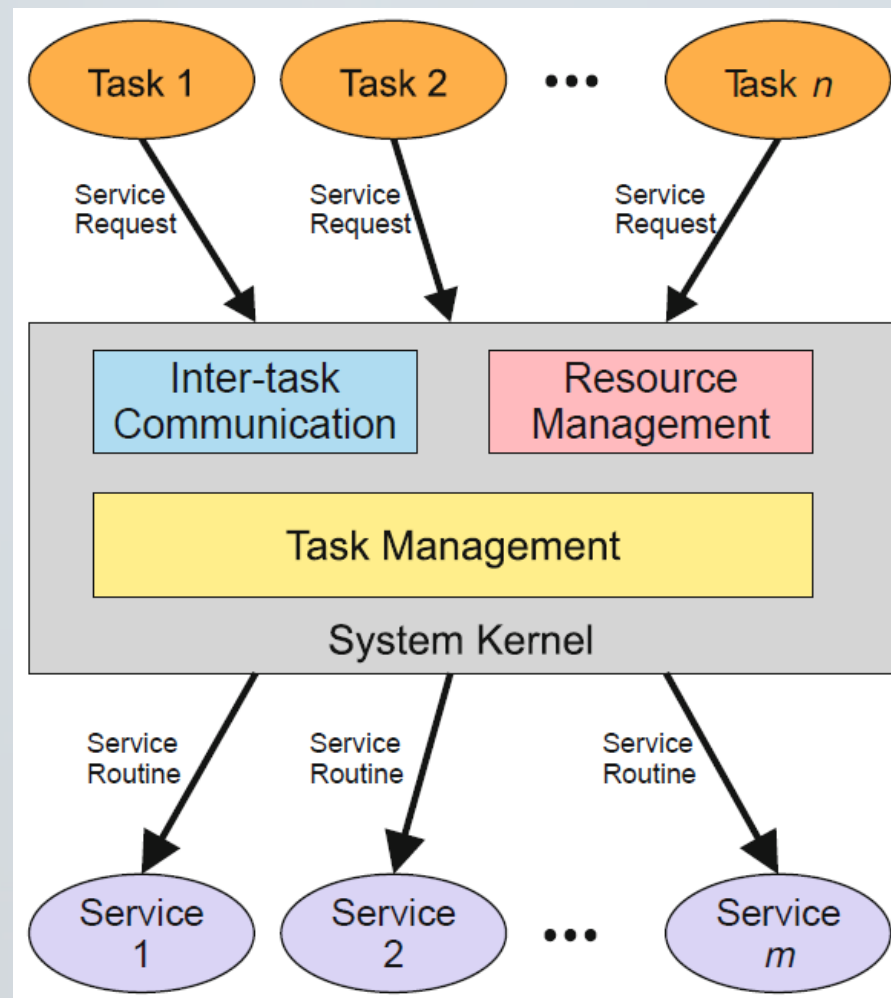
# Softverska komponenta embeded sistema II

- Glavne komponente sistemskog softvera su:
  - **sistemski zadaci (system task)**
  - **sistemski kernel**
  - **servisi**
- Softver embeded sistema podeljen je na skup manjih programa koji se nazivaju **sistemski zadaci**
- Svaki zadatak zadužen je za izvršavanje specifične akcije, za šta koristi neke od **sistemskih resursa**
- Zadaci upućuju zahteve **sistemskom kernelu** kako bi realizovali njima pridružene akcije
- U primeru sa mikrotalasnom rernom rad sistema može se dekomponovati na skup zadataka koji uključuju:
  - čitanje tastature kako bi se odredila akcija koju korisnik želi da izvede,
  - prikaz informacija na sistemskom displeju,
  - uključivanje magnetrona tokom odgovarajućeg perioda vremena, itd.



# Softverska komponenta embeded sistema III

- Softverska komponenta koja **upravlja** korišćenjem raspoloživih sistemskih resursa unutar embeded sistema naziva se **kernel**
- Pod **sistemskim resursima** podrazumevamo sve one komponente sistema koje su neophodne da bi se mogao izvršiti neki zadatak i one uključuju:
  - **sistemsku memoriju**
  - **ulazno/izlazne uređaje**
  - **CPU**
  - **ostale hardverske komponente**
- Kernel prihvata zahteve od sistemskih zadataka koje zatim raspoređuje po prioritetima koje određuje **menadžer zadataka**
- U slučaju da više zadataka zahteva isti resurs, menadžer resursa određuje **politiku korišćenja resursa unutar sistema**

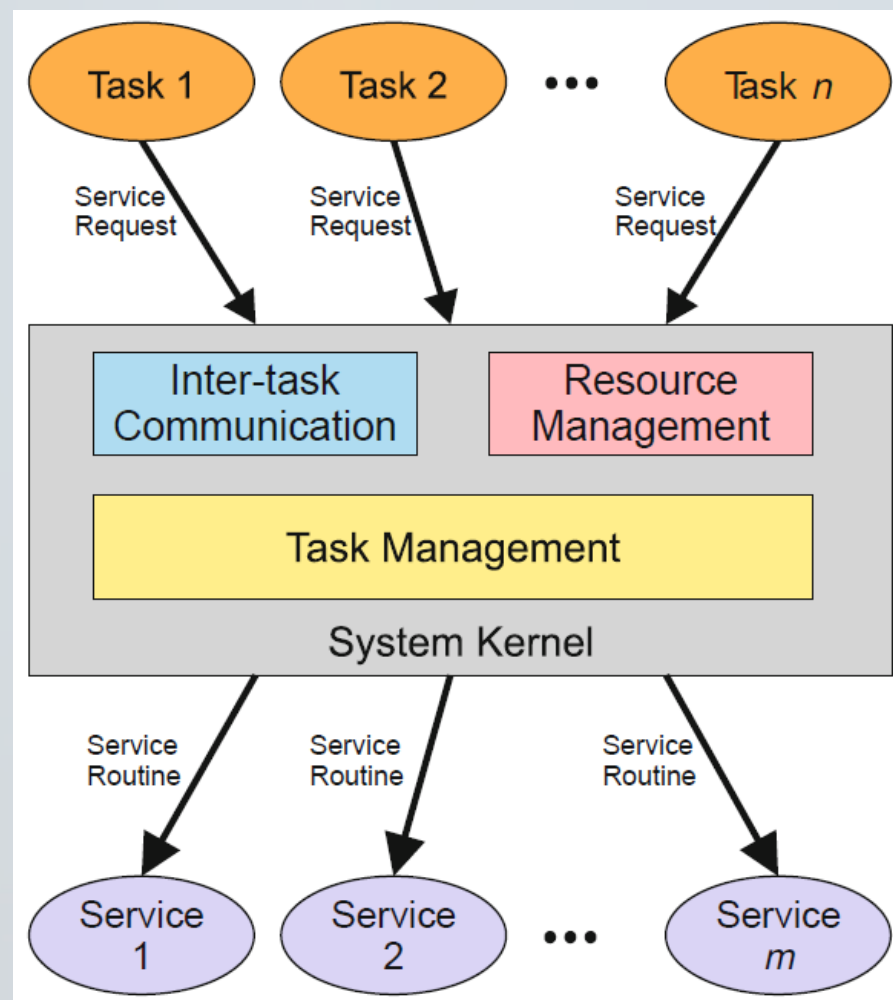


```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```



# Softverska komponenta embeded sistema IV

- U mnogim embeded sistemima javlja se potreba za **kommunikacijom između različitih sistemskih zadataka** sa ciljem razmene informacija između njih
- Kernel obezbeđuje okvir koji omogućava **pouzdanu komunikaciju** između sistemskih zadataka i koordiniše saradnju
- Sistemski zadaci se opslužuju pomoću **servisnih rutina**
- **Servisna rutina** je deo programskog koda koji obezbeđuje funkcionalnost nekog sistemskog resursa
- U nekim embeded sistemima servisne rutine se nazivaju **drajverima**
- Servisne rutine mogu se aktivirati **sistemom prozivke** ili korišćenjem **sistema prekida**



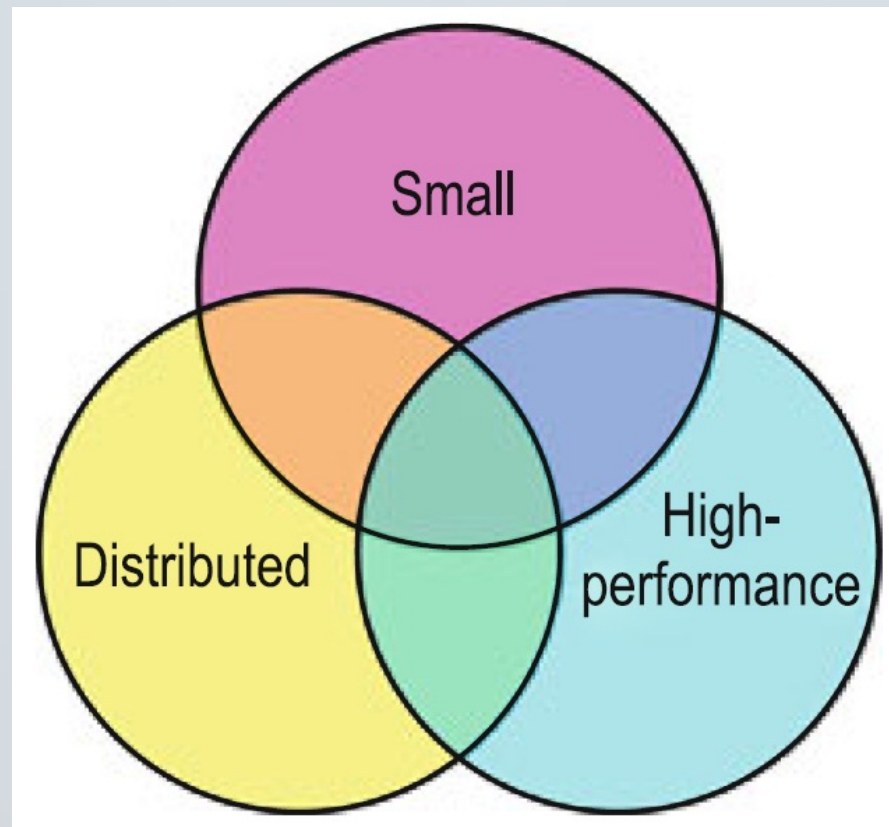
```
entity test_shift is
  generic ( width : integer := 17 );
  port ( clk : in std_ulogic;
        reset : in std_ulogic;
        load : in std_ulogic;
        en : in std_ulogic;
        outp : out std_ulogic );
end test_shift;
```

# Klasifikacija embeded sistema

```
shifter : process (en, reset)
begin
  if ( reset = '0' ) then
    shift_reg <= (others => '0');
  elsif rising_edge ( clk ) then
    if (load = '1' ) then
      shift_reg <= unsigned (inp);
    elsif ( en = '1' ) then
```

# Klasifikacija embeded sistema

- Embeded sistemi mogu se podeliti na sledeće tri kategorije:
  - **Mali embeded sistemi**
  - **Distribuirani embeded sistemi**
  - **Embeded sistemi visokih performansi**
- Potrebno je primetiti da ove kategorije nisu uzajamno isključive
- Postoje “**sive zone**” gde se karakteristike dve ili tri kategorije prepliću te je teško asociirati sistem tačno jednoj kategoriji
- Ipak, u većini slučajeva relativno je jednostavno identifikovati kojoj od kategorija pripada posmatrani embeded sistem



# Mali embeded sistemi

- Ova klasa embeded sistema centrirana je **oko jednog mikrokontrolera** koji upravlja čitavom aplikacijom
- Ovi sistemi su **visoko integrisani**, zahtevajući vrlo mali broj dodatnih analognih komponenti, senzora, aktuatora kao i korisničkog interfejsa
- Rade sa minimalnim ili čak i bez ikakvog održavanja, niske su cene i proizvode se u ogromnim serijama
- Softver u ovim sistemima se tipično sastoji samo od **jednog sistemskog zadatka** i retko zahteva korišćenje nekog operativnog sistema
- Primeri malih embeded sistema su:
  - sistem za kontrolu pritiska u gumama
  - kontroleri u dečijim igračkama, itd.

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

# Distribuirani embeded sistemi

- Kod **distribuiranih embeded sistema**, CPU se nalazi u **odvojenom integrisanom kolu** od ostalih komponenti, kao što su memorija, ulazno/izlazni uređaji, koprocesori
- Ostale specifične funkcije sistema distribuirane su na većem broju odvojenih integrisanih kola koja čine ono što se naziva **procesorski čipset** (processor chipset)
- Iako robusnost nije kritična, kod ovih sistema se zahteva mogućnost održavanja, unapređivanja i dijagnostike
- Sistemi obično opslužuju **više** različitih **zadataka** tako da je korišćenje operativnih sistema uobičajeno
- Primeri distribuiranih embeded sistema su: video procesori, kontroleri video igara, data logeri, mrežni procesori, itd.

```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```

# Embedded sistemi visokih performansi

- U ovu kategoriju spadaju visoko specijalizovani sistemi koji zahtevaju izvođenje **brzih proračuna, robusnost, tolerantnost na greške** i visoki stepen održavanja
- Ovi sistemi obično zahtevaju korišćenje **ASIC kola**, tipično su **distribuirani**, mogu da sadrže DSP i FPGA komponente kao deo osnovne hardverske arhitekture
- U većini slučajeva složenost njihovog softvera zahteva korišćenje operativnih sistema za rad u realnom vremenu (RTOS)
- Proizvode se u malim količinama i njihova cena je visoka
- Ova kategorija embedded sistema sreće se u vojnim i vazdušno-kosmičkim aplikacijama i uključuje: kontrolu letenja, sisteme za navođenje projektila, navigacione sisteme u svemirskim letilicama

```
shift_reg <= unsigned(inp);  
elsif (en = '1') then
```



```
entity test_shift is
  generic ( width : integer := 17 );
  port ( clk : in std_ulogic;
        reset : in std_ulogic;
        load : in std_ulogic;
        en : in std_ulogic;
        outp : out std_ulogic );
end test_shift;
```

## Teorijska pitanja P1

```
shifter : process (en, reset)
begin
  if (reset = '0') then
    shift_reg <= (others => '0');
  elsif rising_edge ( clk ) then
    if (load = '1') then
      shift_reg <= unsigned (inp);
    elsif ( en = '1' ) then
```

# Predavanja 1- Uvod u embeded sisteme 1

## Spisak teorijskih pitanja uz Predavanja 1

1. Definicija embeded sistema i dve glavne karakteristike
2. Struktura savremenog embeded sistema
3. Hardverska komponenta embeded sistema
4. Softverska komponenta embeded sistema
5. Klasifikacija embeded sistema

```
shift_reg <= unsigned (inp);  
elsif ( en = '1' ) then
```

```
entity test_shift is  
  generic ( width : integer := 17 )  
end entity;
```



```
  shift_reg <= unsigned (inp);  
  elsif ( en = '1' ) then
```