Mikroprocesorska elektronika

Predavanje II

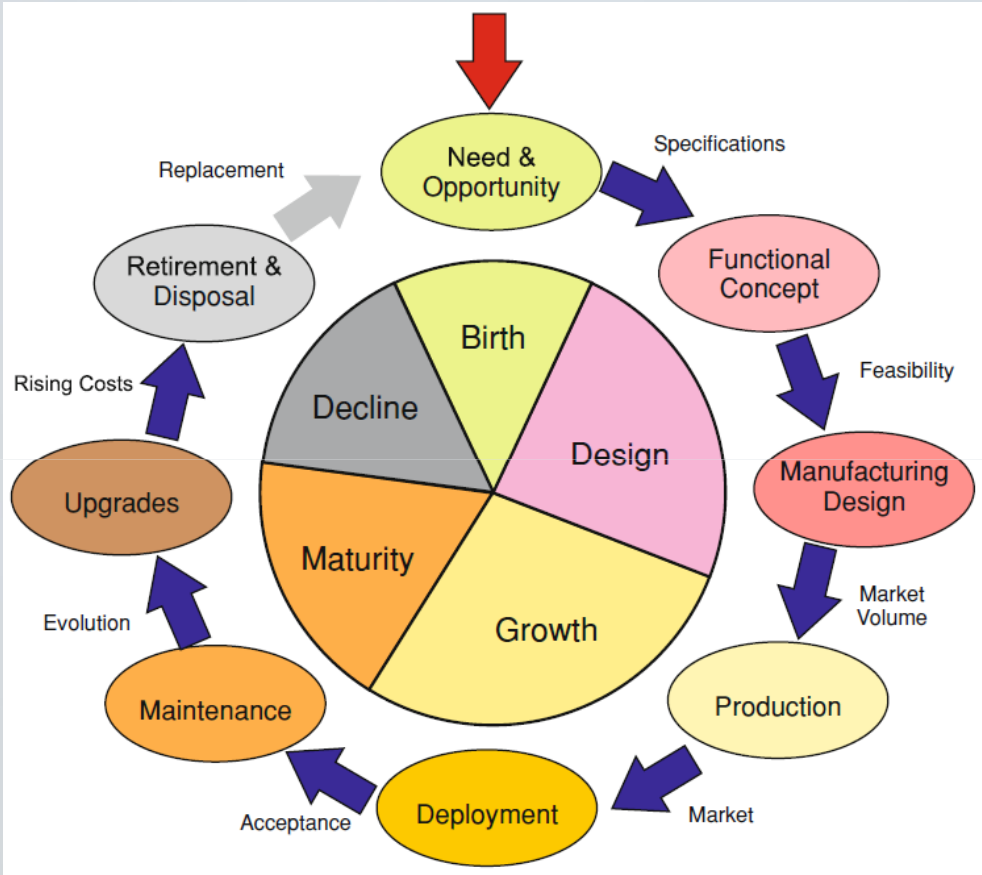
Sadržaj predavanja

- Životni ciklus emebeded sistema
- Ograničenja prilikom projektovana embeded sistema

Životni ciklus emebeded sistema

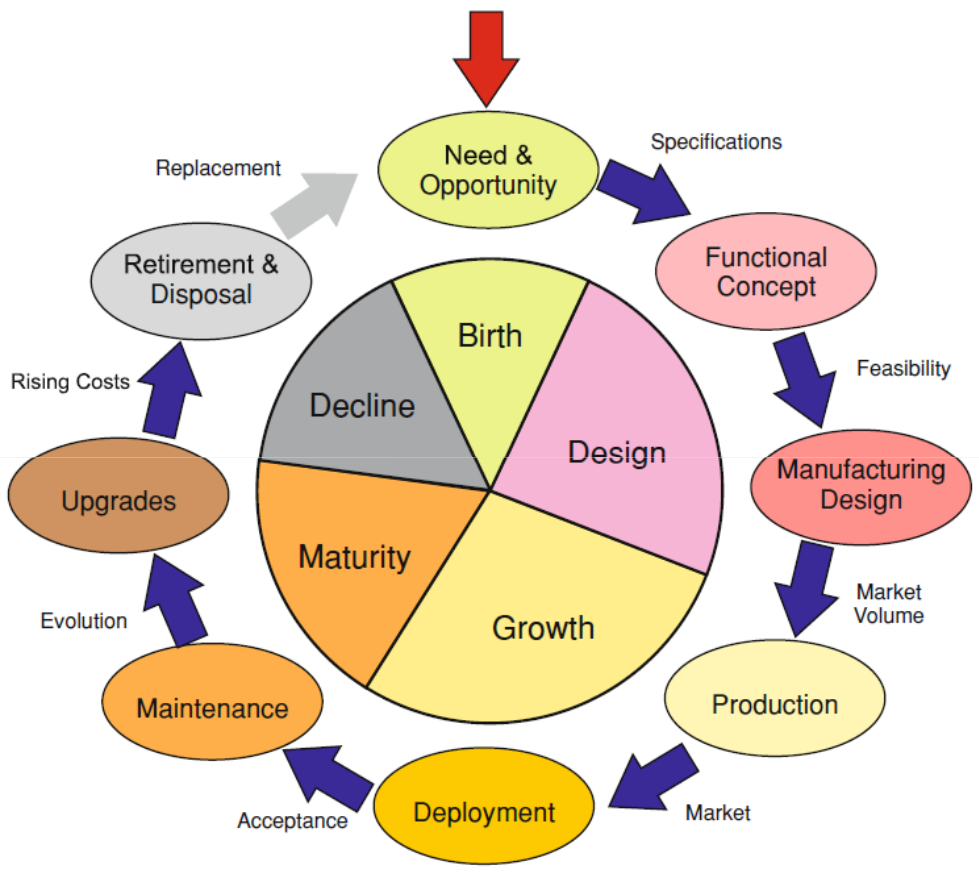
Životni ciklus embeded sistema I

- Embeded sistemi imaju konačni životni vek tokom kojeg prolaze kroz različite faze, počevši od koncepta ili rađanja do odlaganja, i u mnogim slučajevima, ponovnog rađanja
- Faze kroz koje prolazi svaki embeded sistem čine životni ciklus embeded sistema
- U okviru životnog ciklusa embeded sistema mogu se identifikovati pet faza:
  1. Koncepcija ili rađanje
  2. Dizajn
  3. Rast
  4. Zrelost
  5. Opadanje



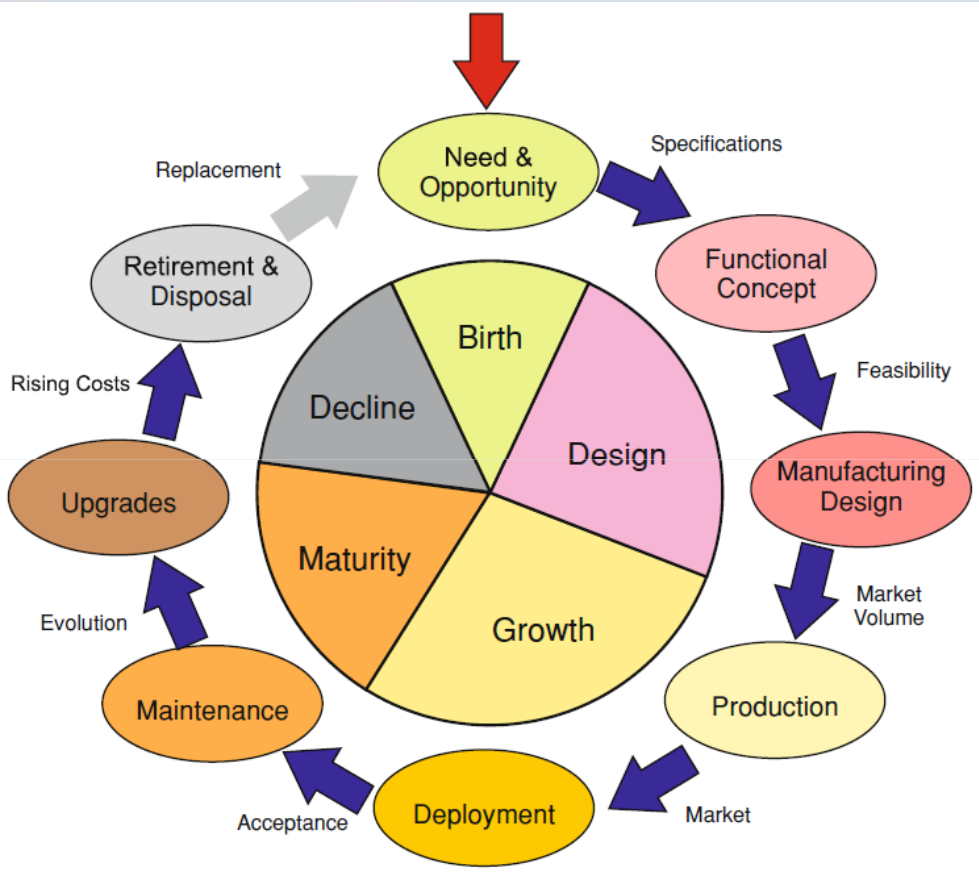
Životni ciklus embeded sistema II

- Embeded sistem započinje svoj životni vek fazom rađanja
- U fazi rađanja embeded sistem prvi put se zamišlja, usled identifikacije potrebe za rešavanjem određenog problema ili uviđanjem mogućnosti da se embeded tehnologija primeni na neki stari problem
- U dizajn fazi, embeded sistem prvo prolazi kroz korak funkcionalnog dizajna, tokom koje se razvijaju “proof-of-concept” prototipovi
- U ovoj fazi vrši se podešavanje funkcionalnosti sistema kako bi on u što boljoj meri odgovarao zahtevima aplikacije za koju se razvija

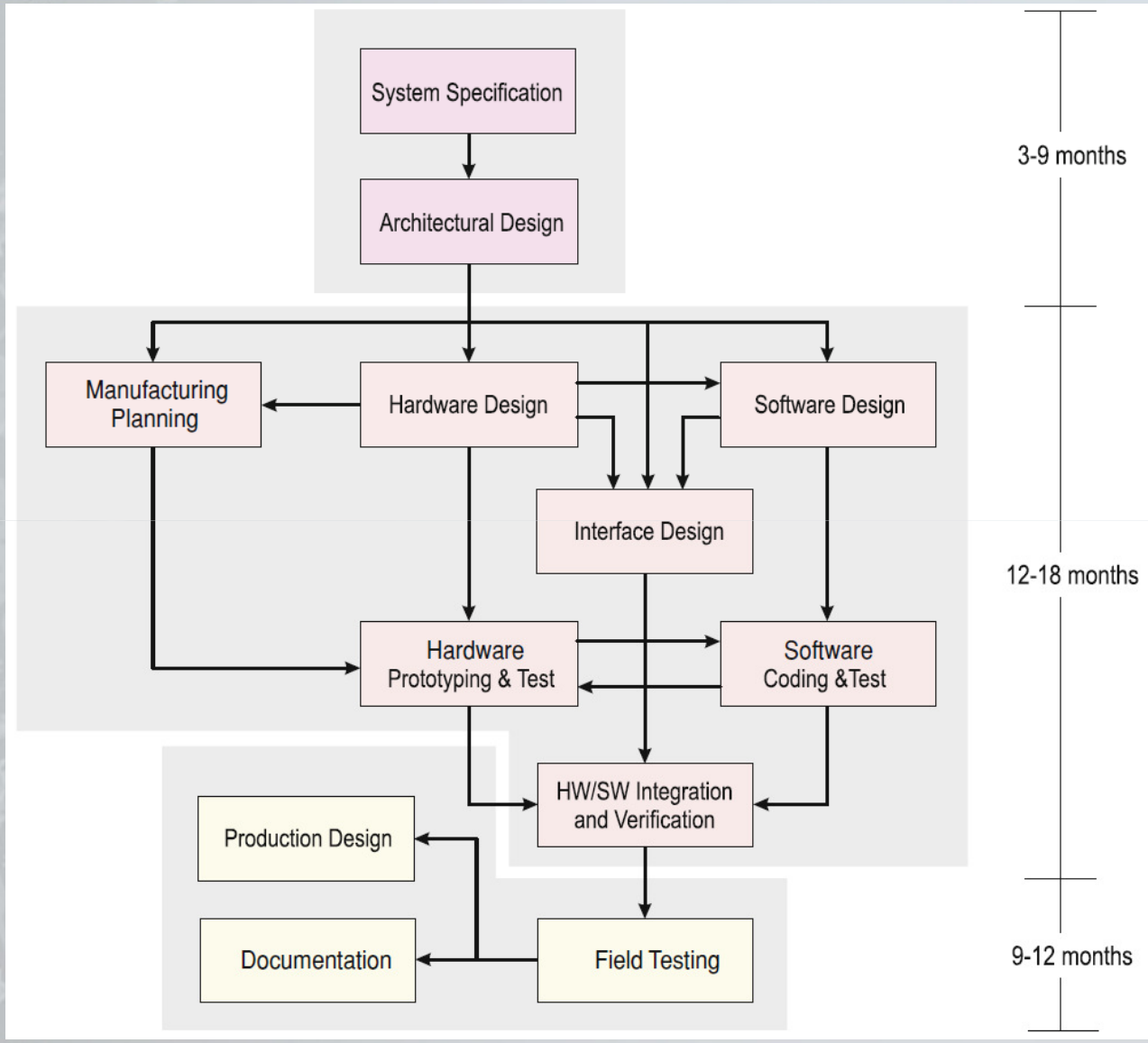


Životni ciklus embeded sistema III

- Studija izvodljivosti je sledeći korak dizajn faze
- Studija izvodljivosti uzima u obzir cenu proizvodnje, prozor na tržištu, zastarivost komponenti, itd., da bi odlučila da li će proizvodnja embeded sistema započeti ili ne
- U ovom drugom koraku, utvrđuje se način na koji će prototip prerasti u proizvod, kao i logistika potrebna za njegovu proizvodnju
- Dizajn faza je po svemu najskuplja faza u životnom ciklusu embeded sistema, obzirom da većina jednokratnih troškova nastaje u ovoj fazi
- Faza rasta započinje proizvodnjom sistema kako bi se zadovoljile očekivane potrebe tržista

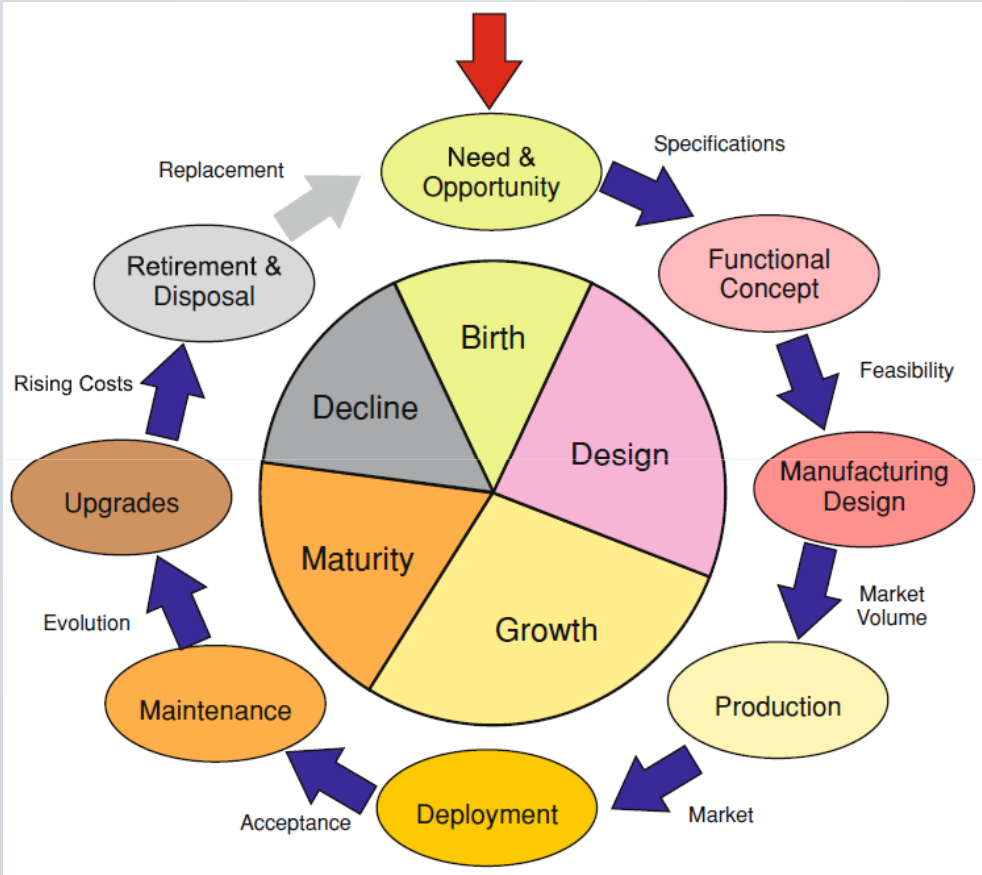


Tipični tok dizajna embeded sistema



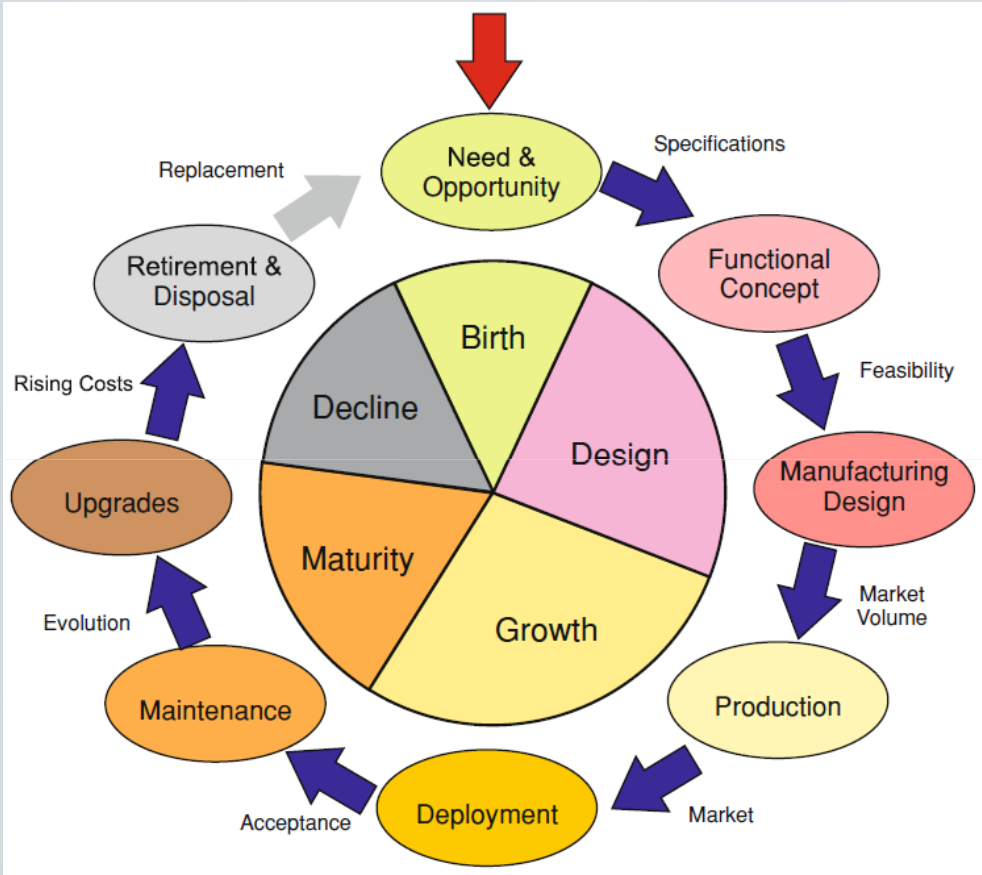
Životni ciklus embeded sistema IV

- Proizvodnja se planira na takav način da se sistem plasira na tržište na početku njegovog “prozora” (market window)
- Proizvodnju sledi korak uvođenja sistema na tržište, koji uključuje distribuciju, instalaciju i pripremu sistema
- U fazi zrelosti embeded sistem se održava funkcionalnim, uz povremena poboljšavanja
- Ova faza uključuje pružanje tehničke podrške, periodičnog održavanja i servisiranja
- Od faze zrelosti se očekuje da omogući eksploataciju proizvoda uz minimalne troškove



Životni ciklus embeded sistema V

- Kako embeded sistem počinje da stari, troškovi njegovog održavanja počinju da rastu, signalizirajući početak faze opadanja
- U fazi opadanja, komponente sistema postaju zastarele te se stoga teško nabavljaju, utičući na porast troškova održavanja, servisiranja i poboljšavanja
- Kada cena održavanja, servisiranja i poboljšavanja postojećeg proizvoda postane veća od cene zamene celog sistema odjednom, nastupa trenutak povlačenja sistema iz upotrebe injegovog odbacivanja
- U mnogim slučajevima, povlačenje jednog proizvoda stvara potrebu za kreiranjem novog dizajna, na taj način započinjući čitav ciklus iz početka



Ograničenja prilikom projektovanja emebeded sistema

Ograničenja prilikom projektovanja embeded sistema I

- Velika većina embeded sistema ugrađuje se u sisteme koji se proizvode u ogromnim serijama
- Ovakvi sistemi su vrlo osetljivi na cenu resursa potrebnih za njihovu proizvodnju
- Pored toga, novi proizvodi moraju da budu završeni, proizvedeni i uvedeni na tržište na vreme koje će rezultovati u maksimizovanju prihoda
- Ova ograničenja utiču na konačni dizajn embeded sistema od početka do kraja njihovog životnog ciklusa

Ograničenja prilikom projektovanja embeded sistema II

- Iako lista ograničenja o kojima je potrebno voditi računa prilikom razvoja embeded sistema zavisi od slučaja do slučaja, ona tipično uključuje:
  - **Funkcionalnost** - sposobnost sistema da ostvaruje funkciju za koju je projektovan
  - **Cena** - količina resursa potrebnih da se osmisli, projektuje, održava i odbaci embeded sistem
  - **Performanse** – sposobnost sistema da ostvari svoju funkciju u zadatom vremenu
  - **Veličina** – fizički prostor koji zauzima krajnje rešenje
  - **Snaga i energija** – zahtevana energija da bi sistem mogao da obavlja svoju funkciju
  - **Vreme razvoja** – vreme potrebno da se sistem od trenutka osmišljavanja ponudi na tržištu
  - **Pouzdanost**– sposobnost sistema da ostane funkcionalan u zahtevanom vremenskom intervalu

Funkcionalnost I

- Od svakog embeded sistema se očekuje da ima funkcionalnost koja rešava problem za koji je projektovan
- Iako se ovaj zahtev može činiti trivijalnim, njegovo ispunjavanje je izuzetno teško
- Funkcionalna verifikacija embeded sistema je vrlo težak problem koji po nekim procenaa oduzima čak do 70% vremena koje stoji na raspolaganju za razvoj sistema
- Problem verifikacije funkcionalnosti hardverskih i softverskih komponenti embeded sistema je NP težak
- Iako u praksi postoje različite metode koje se mogu koristiti za funkcionalnu verifikaciju, ni jedna od njih ne garantuje optimalno rešenje

Funkcionalnost II

- U praksi se stoga najčešće koristi kombinacija više različitih metoda kako bi se minimizovala pojava neočekivanog ponašanja sistema, odnosno sistemskih bagova

- Najčešće korišćene metode funkcionalne verifikacije uključuju:
  - **Bihevijalnu simulaciju** – sistem se opisuje na nivou ponašanja, a zatim se pomoću odgovarajućih alata vrši simulacija ponašanja modela. Ovaj pristup je naročito koristan ukoliko se embeded sistem razvija korišćenjem IP jezgara za koje obično postoje dostupni bihevijalni modeli, omogućavajući da se verifikacija izvrši u ranoj fazi razvoja sistema.

  - **Simulaciju procesora** – ovaj metod koristi programe koji omogućavaju simulaciju funkcionalnosti softvera razvijenog za ciljni procesor ili mikrokontroler na personalnom računaru. Ovi simulatori su vrlo korisni za verifikaciju softvera bez potrebe za korišćenjem kranjeg hardvera na kome će se softver izvršavati.

Funkcionalnost III

- **JTAG debagere** – JTAG standard omogućava standardizovani pristup unutršnjim delovma digitalnih elektronskih kola pomoću jedinstvenog test porta. Mnogi razvojni sistemi pružaju mogućnost debagovanja procesora preko JTAG porta.

- **Hardverske emulatore** – hardverska emulacija je proces oponašanja ponašanja jedne hardverske komponente pomoću druge sa ciljem olakšanog debagovanja.

Cena I

- Razvoj embeded sistema je vrlo skup proces
- U generalnom smislu, ukupna cena proizvodnje C_T, V jedinica embeded sistema može se izračunati pomoću sledećeg izraza

$$C_T = NRE + (RP \cdot V)$$

- gde *NRE* predstavlja jednokratne troškove proizvodnje, *RP* varijabilne troškove proizvodnje, a *V* broj jedinica proizvoda koji se proizvodi
- Jednokratni troškovi proizvodnje obuhvataju sve troškove koji su vezani za proces projektovanja i testiranja embeded sistema, kao što su:
  - Istraživanje, planiranje, projektovanje hardvera i softvera i njihova integracija
  - Verifikacija, testiranje prototipova, planiranje proizvodnje, dokumentovanje dizajna

Cena II

- Varijabilni troškovi uključuju sve troškove neophodne da bi se proizvela jedna jedinica embeded sistema i uključuju:
  - Cenu komponenti, ploča, kućišta i pakovanja neophodnih da bi se prozveo jedan embeded sistem
  - Cenu testiranja, a ponekad i cenu održavanja, poboljšavanja i odlaganja sistema
- Varijabilni troškovi zavise i od obima proizvodnje i uvek su značajno manji od jednokratnih
- NRE troškovi mogu biti izraženi u milionima eura, dok RP troškovi mogu biti reda nekoliko eura, pa čak i euro centi.

Cena III

- Na primer, razmotrimo troškove razvoja prvog iPod uređaja kompanije Apple
- Razvoj koncepta, definisanje specifikacije i arhitekture uređaja, dizajn hardvera i softvera, debagovanje sistema, planiranje izgleda kućišta, planiranje proizvodnje, tehnička podrška zahtevao je rad od skoro 300 inženjera tokom jedne godine
- Konzervativna procena jednokratnih troškova razvoja iPod uređaja kreće se između 5 i 10 miliona dolara
- Varijabilni troškovi proizvodnje prve generacije iPod uređaja procenjuju se na 120 dolara
- Apple je prijavio prodaju 125.000 uređaja u prvom kvartalu 2002, tako da se u prvoj aproksimaciji ova brojka može koristiti i za broj fabrikovanih urešaja, V

Cena IV

- Uzimajući prethodne brojke u obzir, uz pretpostavku da su jednokratni troškovi iznosili 10 miliona dolara, dolazimo do cene od 25 miliona dolara za proizvodnju prvog kontigenta iPod uređaja
- Ako je potrebno 25 miliona dolara za proizvodnju prvog kontigenta iPod uređaja, kako je moguće da su se oni prodavali po ceni od 300 dolara po komadu?
- Odgovor na ovo krije se u količini proizvedenih uređaja

Cena V

- Uobičajena mera cene uređaja jeste cena po jedinici, U_C, koja se dobija kada se ukupna cena proizvodnje podeli sa brojem fabrikovanih jedinica

$$U_C = \frac{C_T}{V} = \frac{NRE}{V} + RP$$

- Prethodna jednačina pokazuje da se jednokratni troškovi distribuiraju na svaku od fabrikovnih jedinica, čineći tako cenu po jedinici proizvoda prihvatljivom za krajnjeg korisnika

- Na primeru iPod uređaja, estimirana jedinična cena proizvodnje iznosi oko 200 dolara, što znači da je prodajna cena od 300 dolara sasvim dovoljna da pokrije troškove proizvodnje

Performanse I

- Performanse sistema najčešće se odnose na sposobnost sistema da obavi svoju funkcionalnost u predviđenom intervalu vremena
- Ponekad, pod performasama podrazumevamo zahtevanu potrošnju električne energije, zahtevanu količinu memorije ili čak cenu
- U nastavku mi ćemo pod performansama sistema podrazumevati njegovu sposobnost da obavi svoju funkcionalnost u predviđenom intervalu vremena
- Imajući ovo u vidu, neko bi mogao pomisliti da što je viša učestanost sistemskog takta, bolje će biti i njegove performanse
- U generalnom slučaju ovo međutim ne mora da bude slučaj, jer performanse embeded sistema zavise od mnogo faktora, od kojih su neki povezani sa hardverskim a neki sa softverskim komponentama

Performanse II

- Hardverski faktori koji utiču na performanse sistema su:
  - **Učestanost sistemskog kloka** – imajući dva identična sistema, koji izvršavaju isti softver, promena učestanosti sistemskog kloka imaće direktan uticaj na vreme potrebno da se završi neki zadatak.
  - **Mikroarhitektura procesora** – korišćenje procesora sa različitom mikroarhitekturom (višestrukim jedinicama za izvršavanje instrukcija, protočnom obradom, skrivenom memorijom, strukturom instrukcija (CISC ili RISC) može u značajnoj meri uticati na vreme izvršavanja programa
  - **Kašnjenje individualnih komponenti** – vreme koje je potrebno da neka komponenta reaguje na promenu na njenim ulazima može u velikoj meri da utiče na performanse čitavog sistema. Tipičan primer predstavlja uticaj kašnjenja memorijskog pristupa na ukupne performanse sistema. Vreme potrebno da memorija odgovori na zahteve upisa ili čitanja direktno utiče na to kojom brzinom procesor može da razmenjuje podatke sa memorijom. Ovo sa druge strane direktno utiče na propusnu moć memorijskog podsistema, a samim tim i na krajnje performanse čitavog embeded sistema.

Performanse III

- **Komunikacioni protokoli** – način na koji transakcije započinju i završavaju kao i koliko često se moraju inicirati u velikoj meri će uticati na vreme koje je embeded sistemu potrebno da završi neki zadatak. Na primer posmatrajmo uređaj kod koga je za svaki prenos jednog podatka neophodno poslati zahtev, čekati na potvrdu da je zahtev primljen, izvršiti sam transfer podatka i na kraju čekati na potvrdu da je podatak uspešno primljen funkcionisati će znatno sporije od uređaja kod kojega je moguće preneti veći broj podataka pri uspostavljanju svake sesije.

- **Modovi sa niskom potrošnjom** – kada neka komponenta ode u stanje niske potrošnje ili u režim mirovanja, njeno ponovno buđenje po pravilu zahteva veliki broj taktova sistemskog kloka, utičući na njeno vreme odziva, a samim tim i na vreme izvršavanja nekog zadatka u embeded sistemu.

- Bitna stvar koju treba zapamtiti je da je velika brzina skupa; što je veća brzina harvera to je on skuplji i njegova potrošnja je veća

- Brzina projektovanog harvera treba da tačno odgovara potrebnoj brzini obrade

Performanse IV

- Softverski faktori takođe utiču na performanse sistema

- Za razliku od hardvera, projektanti se uvek trude da softver učine što je moguće bržim, jer će to rezultovati u efikasnijem sistemu ne samo u pogledu njegovih perofmansi već i u pogledu potrošnje i hardverskih zahteva, na taj način smanjujući ukupne troškove razvoja

- Softverski faktori koji utiču na performanse embeded sistema su:
  - **Algoritamska složenost** – ovo je najverovatnije najvažniji softverski faktor koji utiče na performanse sistema. Pod algoritamskom složenošću podrazumevamo koliko je koraka (ređe resursa) potrebno da bi se kompletirao neki zadatak. Svaki dodatni korak ne samo da direktno utiče na produženje vremena da se završi zadatak već takođe povećava potrošnju i opterećuje procesor. Ukoliko je algoritamska složenost softvera manja on će raditi brže, poboljšavajući ukupne performanse sistema.

Performanse V

- **Redosled izvršavanja zadataka** – odnosi se na određivanje redosleda izvršavanja zadataka u sistemima sa više zadataka (multitasking sistemi). Izabrani redosled može u velikoj meri da utiče na ukupne performanse sistema.

- **Komunikacija između zadataka** – odnosi se na mehanizme koji se koriste za razmenu informacija i delenje sistemskih resursa između sistemskih zadataka. Od dizajna ovog komunikacionog mehanizma zavisi količina podataka koja se mora tazmeniti prilikom komunikacije, samim tim utičući i na vreme potrebno da se ova komunikacija kompletira.

- **Stepen paralelizma** – odnosi se na sposobnost softvera da iskoristi resurse koji dozvoljavaju istovremeno korišćenje. Na primer, ukoliko embeded kontroler sadrži konvekcionalnu ALU jedinicu i harverski množač on ima kapacitet da istovremeno izvrši dve aritmetičke operacije ukoliko jedna od njih predstavlja množenje. Dvojezgarni procesor može da obrađuje informacije simultano u oba jezgra. Na softveru je da na najbolji način iskoristi ovaj raspoloživi paralelizam koji stoji na raspolaganju kako bi skratio vreme izvršavanja nekog zadatka.

Potrošnja I

- Potrošnja u embeded sistemima postala je jedno od glavnih ograničenja, ne samo u prenosivim, baterijski napajanim sistemima
- Prosečna potrošnja sistema definiše se kao brzina kojom sistem troši električnu energiju
- U baterijski napajanim uređajima ova brzina direktno utiče na vreme koje je sistemu potrebno da potpuno isprazni bateriju, odnosno na njegovo vreme autonomije
- Međutim, pored života baterije, potrošnja utiče i na mnoge druge karakteristike embeded sistema

Potrošnja II

- **Pouzdanost** – U digitalnim kolima, koja rade na relativno stabilnim naponima napajanja, snaga koja se troši u sistemu direktno se prenosi na intezitet struje koja cirkuliše kroz sistem (njegove komponente i veze kojima su one povezane). Ovo proticanje struje uzrokuje toplotnu disipaciju, fizički stres komponenti kao i pojavu šuma, što sve zajedno utiče na pouzdani rad sistema.

- **Potreba za hlađenjem** – velika potrošnja električne energije utiče na veliko zagrevanje sistema. Ova toplota se mora na neki način ukloniti iz sistema, kako bi se obezbedio njegov pouzdan rad. Povećana potrošnja može dovesti do potrebe da se pređe sa pasivnih tehnika hlađenja pomoću hladnjaka na aktivne tehnike koje uključuju ventilatore, tečne rashladne sisteme, itd. Potreba za korišćenjem ovih naprednijih tehnika hlađenja dovodi do povećanja cene proizvodnje kao i povećanja fizičkih dimenzija i težine porjektovanog uređaja.

- **Način projektovanja sistema za napajanje** – povećana potrošnja zahteva projektovanje snažnijih sistema za napajanje koji više ne mogu biti bazirani na baterijama ili drugim alternativnim izvorima energije. Pored toga ovo će takođe povećati troškove proizvodnje i veličinu/težinu projektovanog uređaja.

Potrošnja III

- Minimizaciji potrošnje električne energije unutar embeded sistema može se pristupiti na razne načine:
  - **Odgovarajućim projektovanjem hardvera** – korišćenjem procesora i hardverskih komponenti koje su posebno projektovane tako da minimizuju potrošnju. Nisko-naponske periferije sa niskom potrošnjom, procesori sa standby i sleep modovima, memorije se niskom potrošnjom su neke od najvažnijih tipova komponenti pomoću kojih je moguće redukovati potrošnju električne energije.

  - **Odgovarajućim projektovanjem softvera** – način na koji je projektovan softver može da ima veliki uticaj na potrošnju sistema. Posmatrana funkcija može se implementirati u softveru na razne načine. Ukoliko se izvrše merenja potrošnje električne energije prilikom izvršavanja ovih različitih implementacija (power/energy profiling), može se izabrati implementacija koja je najoptimalnija po pogledu potrošnje. Generalno, implementacije koje u manjem broju ciklusa kompletiraju izvršavanje nekog zadatke, imaće manju potrošnju.

Vreme razvoja

- Vreme razvoja (Time to Market) definiše se kao vreme koje je potrebno da se novi embeded sistem od faze koncepta dovede to faze u kojoj je spreman da se ponudi na tržištu
- Vreme razvoja postaje jedno od kritičnih ograničenja za sisteme koji imaju uzak prozor na tržištu (Market Window)
- Prozor na tržištu definiše se kao maksimalno vreme tokom kojeg se neki proizvod može uspešno prodavati na tržištu
- Ukoliko novi proizvod kasno uđe na tržište, odnosno promaši svoj prozor na tržištu, povrat uloženih sredstava biće minimizovan, čak do te mere da u potpunosti isključi finansijsku isplativost čitavog projekta

Pouzdanost I

- Pouzdanost sistema može se definisati kao njegova sposobnost da garantuje korektnu funkcionalnost u nekom unapred definisanom intervalu vremena
- Sistemi sami po sebi nisu pouzdani; pouzdanost sistema mora se planirati u ranoj fazi razvoja sistema
- Pouzdanost sistema može biti od veće ili manje važnosti, u zavisnosti od tipa embeded sistema koji se projektuje
- Mali, jeftini, sistemi kratkog života, kao što su mobilni telefoni i sitni kućni aparati, se menjaju novijim, savremenijim uređajima pre nego se njihova pouzdanost naruši
- Veliki, skupi, složeni embeded sistemi, kao što su aero-kosmički sistemi, medicinski sistemi, sigurnosni sistemi, moraju da korektno funkcionišu tokom dugotrajnog perioda vremena, zahtevajući visoku pouzdanost

Pouzdanost II

- U opštem slučaju, pouzdanost sistema mora da obezbedi načine da sistem može da ostvaruje četiri osnovna tipa akcija:
  - **Korektivne akcije** – da bi se omogućila popravka kvarova koji su detektovani u sistemu
  - **Adaptivne akcije** – koje omogućavaju uvođenje funkcionalnih modifikacija koje obezbeđuju da sistem funkcioniše i u promenljivim sredinama
  - **Unapređujuće akcije** – koje dozvoljavaju dodavanje poboljšanja sistema kao rezultat novih propisa i zahteva
  - **Preventivne akcije** – koje predviđaju i sprečavaju pojavu uslova koji bi mogli narušiti normalno funkcionisanje sistema


entity test_shift is
generic (width : integer := 7);
shift_reg <= unsigned (inp);
elsif (en = '1') then