Sveprisutno računarstvo

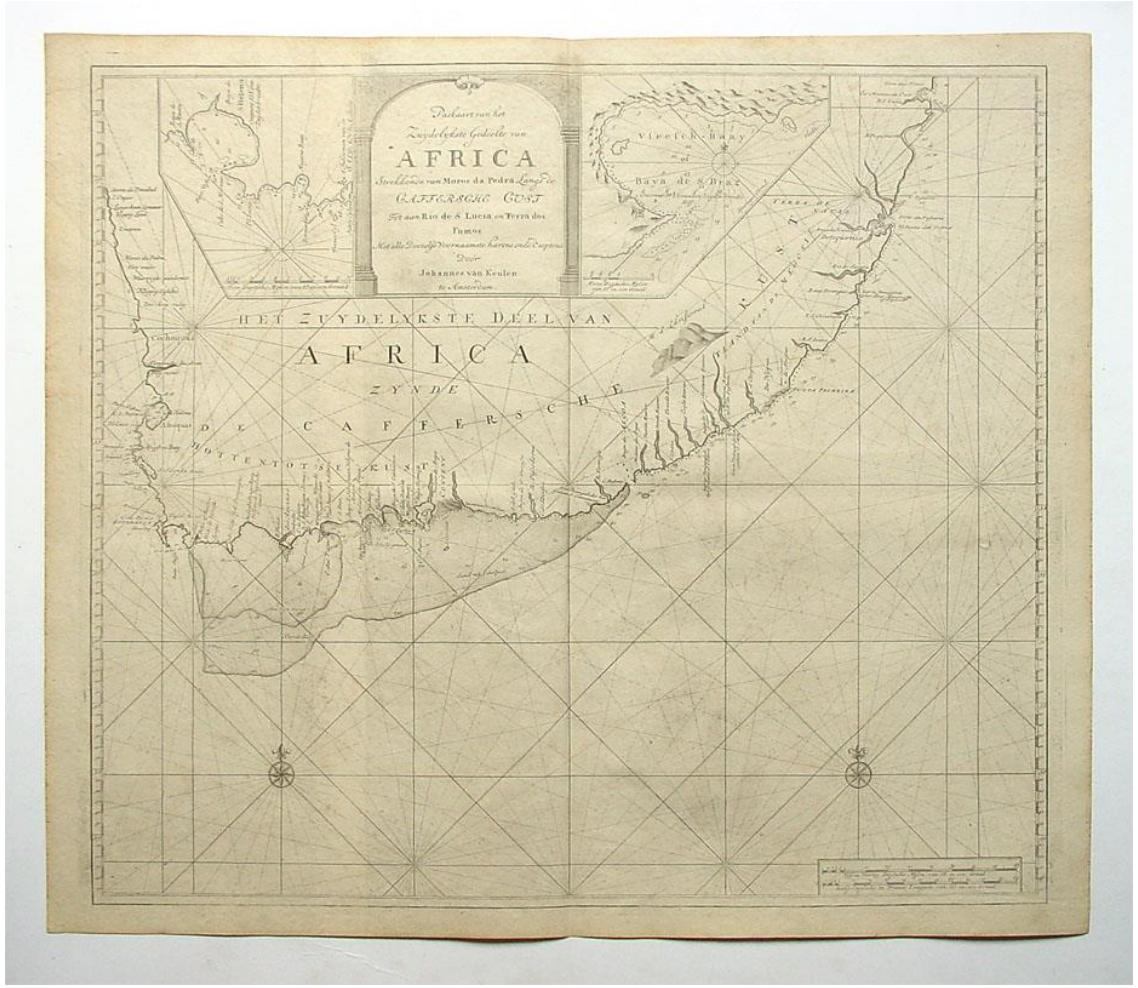
4. Vrijeme

- Uloga vremena u računalnom sustavu
- Signal vremenskog vođenja
- Vremenski sklopovi
- Rad u stvarnom vremenu

Creative Commons

Uloga vremena u računalnom sustavu

Značenje mjerenja vremena



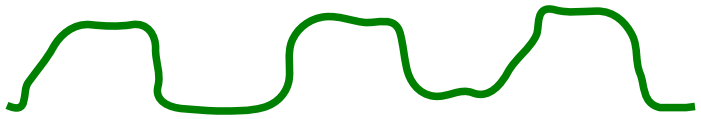
Vrijeme u (sveprisutnom) računalnom sustavu

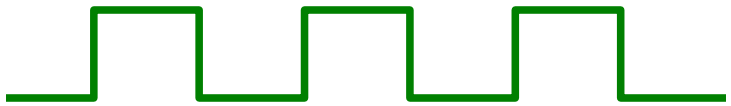
- Signal vremenskog vođenja – sinkronizacija rada komponenti sustava
- Mjerenje protoka vremena u sustavu (i okolini)
- Obavljanje zadataka u ograničenom vremenu, diktiranom od okoline

Signal vremenskog vođenja

Takt

- Signal takta (vremenskog vođenja)
  - Koristi se za sinkroniziranje rada sklopova
    - Izvođenje operacija točno određenim redoslijedom
  - Interni signal takta ne mora odgovarati vanjskom
    - Takt se prilagođava potrebi
      - Frekvencija se množi, dijeli
      - Mijenja se odnos aktivne i neaktivne poluperiode (popunjenost)
      - Mijenja se valni oblik (najčešće ipak pravokutni signal)

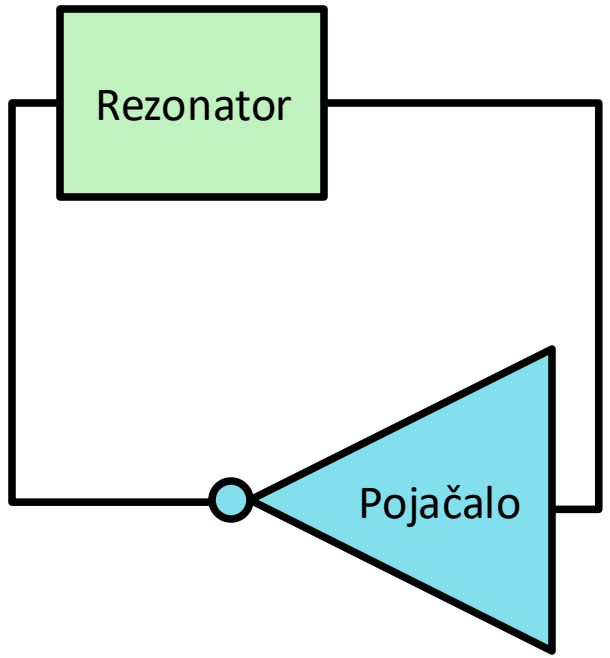




- Moguće su i složene strukture bez signala takta
  - Procesorske jezgre, veze između njih, ...
- Sinkronizacija taktom prevladava

Osnovni sklopovi

- Generator signala takta
  - Stvara signal takta (vremenskog vođenja)
  - Osnovni dijelovi oscilatora
    - Rezonator
    - Pojačalo



- Sklopovi za razvođenje signala takta
  - Problem kašnjenja signala po linijama

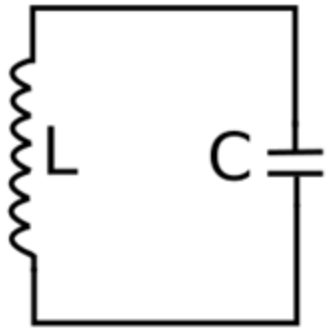
Rezonator

- Sklop koji može *titrati* ako mu se dovodi energija
- U kombinaciji s pojačalom tvori oscilator
  - Pojačalo ostvaruje povratnu vezu kako bi se održale oscilacije
  - Pojam *faktora dobrote* (*quality factor*, *Q-factor*)
    - Omjer pohranjene i izgubljene energije u svakom ciklusu
- Mogući su analogni i digitalni oscilatori
  - Analogni
    - Fizikalni princip njihala
    - Energija pohranjena u dva krajnja stanja njihala
  - Digitalni
    - Oscilator mijenja stanje između '0' i '1'

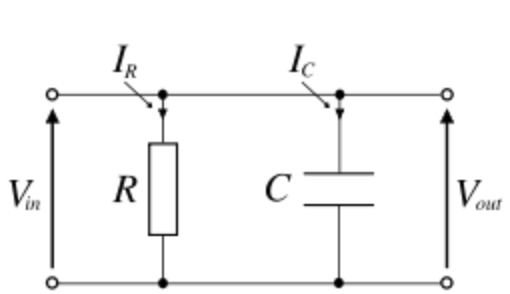
Rezonatori i oscilatori

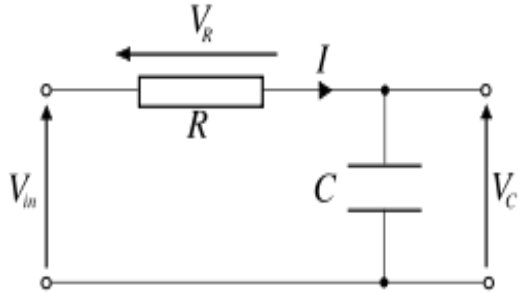
- Analogni
  - (R)LC
  - RC
  - Keramički rezonator
  - Kristal
  - Atomski rezonator (rubidij, cezij)
- Digitalni
  - XOR
  - Prstenasti oscilator
- Oscilatori (kao komponente)
  - Stabilizirani elektronički oscilator
  - PLL (*Phase-locked loop*)

LC i RC sklopovi



$$f = \frac{1}{2\pi\sqrt{LC}}$$





$$f = \frac{1}{2\pi RC}$$

- Jednostavan sklop, sastoji se od zavojnice i kondenzatora
- Energija u dva krajnja stanja
  - Električno polje u kondenzatoru
  - Magnetsko polje u zavojnici

- Sastoji se od otpora R i kondenzatora C
- Energija pohranjena
  - Električno polje u kondenzatoru
  - Struja/napon na otporniku (ovisno o izvedbi)
  - Sinusoidni i ne-sinusoidni valni oblici

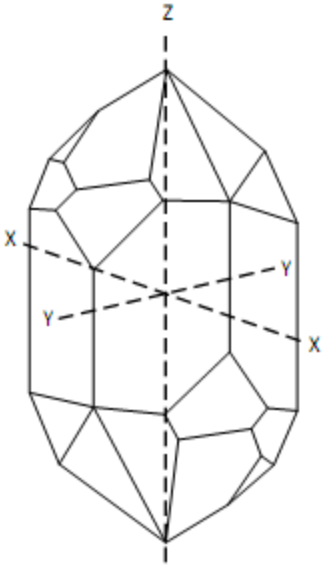
Keramički i kvarcni rezonatori

- Piezo-električni rezonatori
  - Nekih materijali uslijed mehaničkog pritiska proizvode električni naboj
  - Kristali, neke vrste keramike, kosti, ...
- Frekvencijske karakteristike ovise o mehaničkim
  - Rezonantna frekvencija se može odrediti načinom rezanja kristala
    - Ovisi o veličini, obliku, elastičnosti, temperaturi i starenju
  - Kvarc
    - Stabilne karakteristike za većinu primjena

Kristalni rezonatori

$f \approx \frac{1}{debljina}$

- Kristal kvarca
  - Čišćenje
  - Rezanje po kristalnim osima
  - Oblikovanje
  - Starenje
  - Zatvaranje u kućište
  - Testiranje

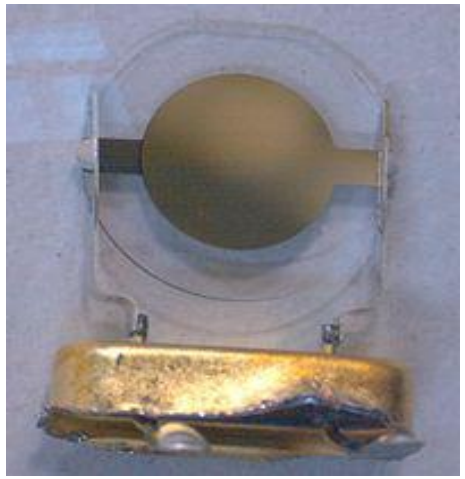


- Frekvencija

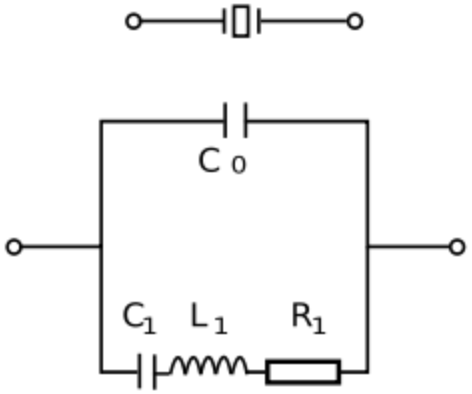
$$f \approx \frac{1}{debljina}$$

Piezoelektrična svojstva kristala

- Dovođenje napona
  - Kristal se izobliči
- Uklanjanje napona
  - Kristal se vraća u prvobitno stanje
  - Stvara električno polje – napon



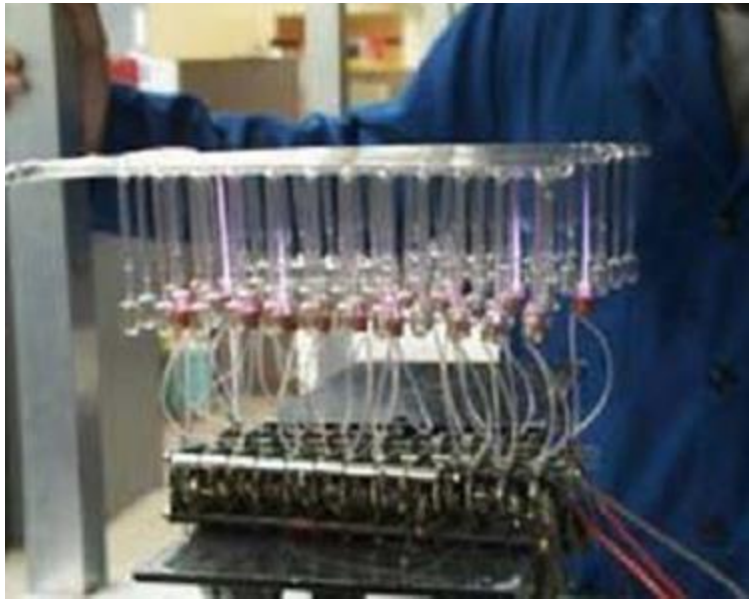
- Ponašanje kao RLC-krug vrlo precizne rezonantne frekvencije
- Kristal ima serijsku i paralelnu rezonanciju
  - Može raditi u više načina rada
  - Primjenjiv za razne obitelji sklopova



Izvor: Wikipedia

Atomski rezonatori

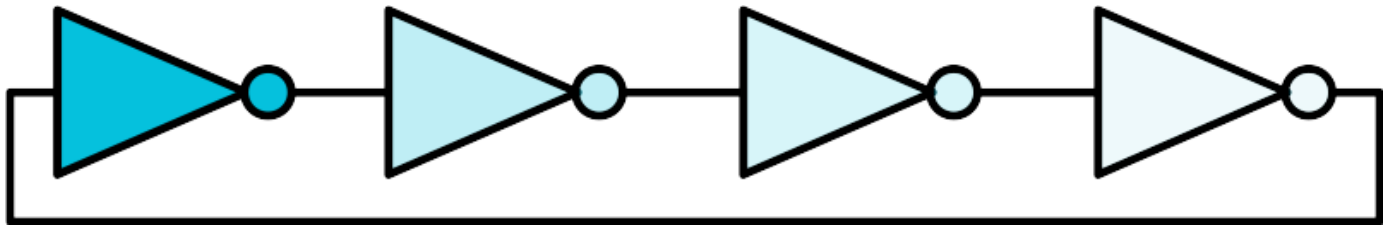
- Visoka stabilnost frekvencije
- Atom prelazi iz stanja više u stanje niže energije
- Svojstvo atoma, ne materijala
  - Fizikalna konstanta
- Problem
  - Atomi nisu slobodni
    - Sudari, električna i magnetna polja
  - Atomi se kreću
    - Toplinski šum
  - Promatranjem atoma utječemo na njihovo stanje



Izvor: Frequency Electronics, 2004.

Digitalni rezonatori

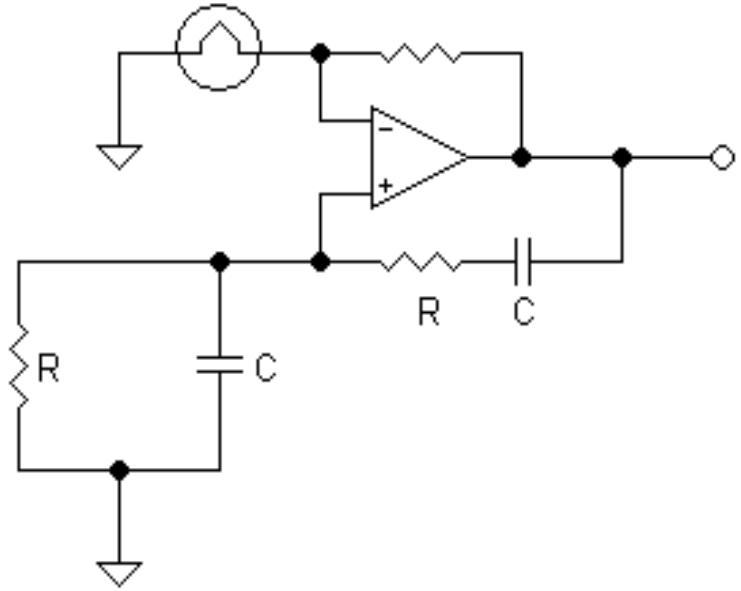
- XOR sklop
- Prstenasti rezonator (oscilator)
  - Niz NOT sklopova spojenih u prsten (paran broj NOT)
  - Svaki sklop se sastoji od n- ili p- MOSFET tranzistora
    - Promjena stanja traje neko vrijeme (kapacitet MOSFET-a)
    - Efekt osciliranja - zbog kašnjenja u propagaciji signala
    - Potrebno dodatno pojačalo
  - Osnovni sklop za testiranje karakteristika poluvodičkog procesa



Izvedbe oscilatora

- Najjednostavniji oscilatori – RC
- RC kao dio mosta
- Najčešće izvedbe sinusoidnih RC oscilatora
  - Wien-ov most
  - "Twin-T"

- Wienov most :
  - 3 otpornika, 2 kapaciteta i operacijsko pojačalo



Izvor: Wikipedia

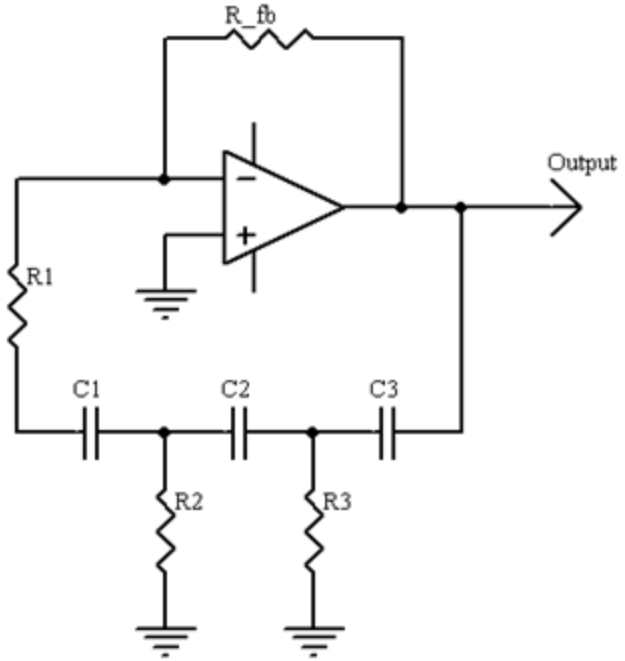
RC - oscilatori

- "Twin-T" oscilator
  - Dva "T"-RC (R-C-R) kruga spojena paralelno
  - Zajedno tvore most koji se može ugoditi na željenu frekvenciju oscilacije

- Oscilator s faznim pomakom
  - Invertirajuće pojačalo i

    povratni filter za pomak faze

    za 180° na frekvenciji osciliranja

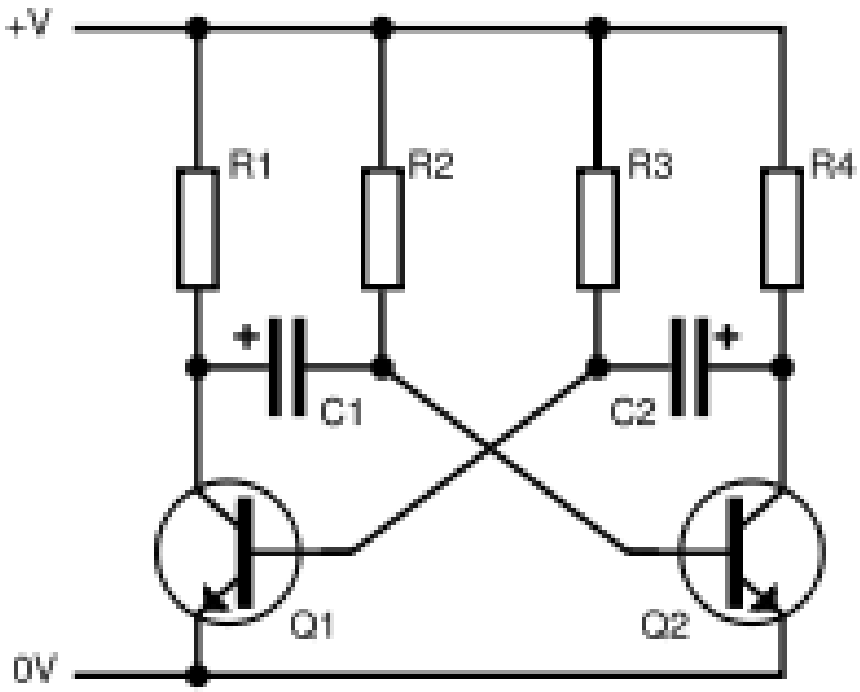


Izvor: Wikipedia

RC - oscilatori

- Ne-sinusoidni valni oblik
  - Najčešće kvadratni valni oblik
  - Multivibratorski sklopovi

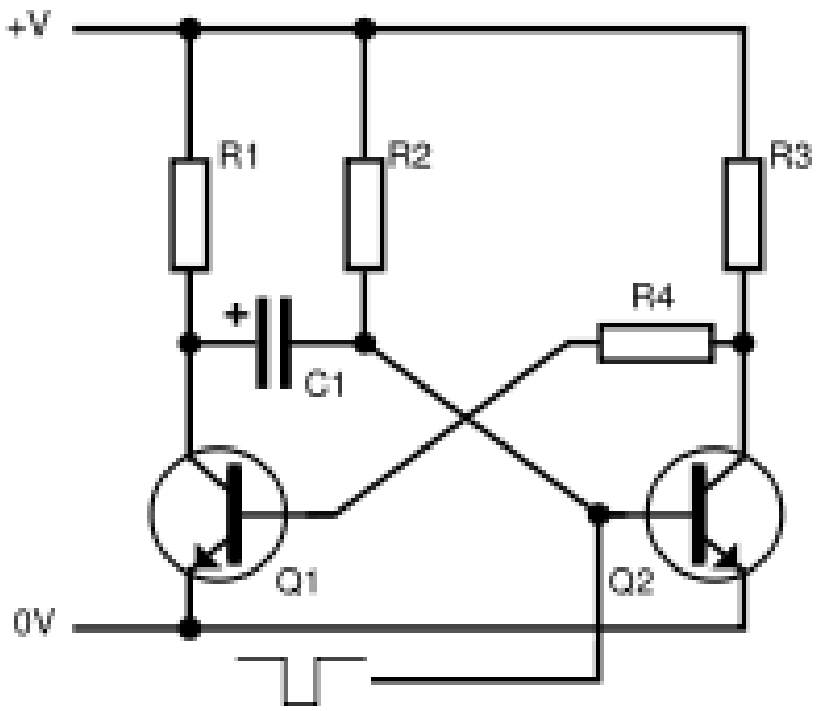
- Multivibratorski sklopovi
  - Astabil
    - Nema stabilnog stanja
    - Sklop oscilira između dva stanja



Izvor: Wikipedia

Multivibratorski sklopovi

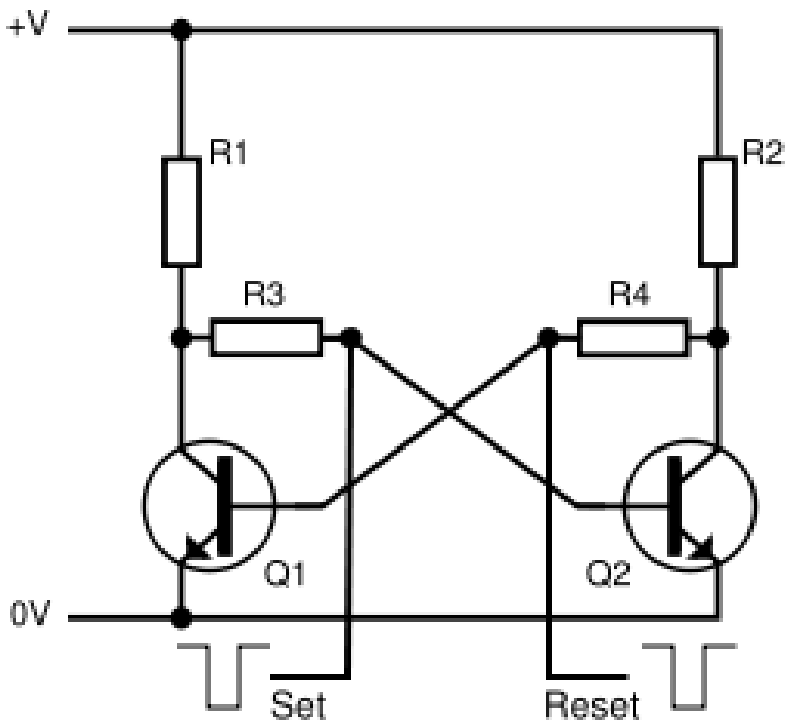
- Monostabil
  - Jedno stabilno stanje
  - Određeni vremenski period je u nestabilnom stanju, a zatim prelazi u stabilno stanje (*one shot circuit*)
  - Pogodan za stvaranje fiksnog vremenskog perioda kao reakcije na vanjski događaj



Izvor: Wikipedia

Multivibratorski sklopovi

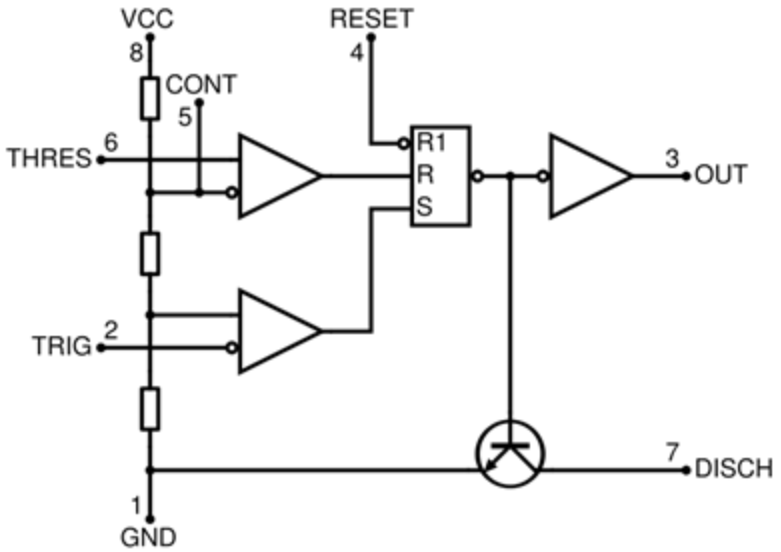
- Bistabil
  - Dva stabilna stanja
  - Sklop je u jednom od stanja neodređeno vrijeme
  - Može se prebacivati iz jednog stanja u drugo pomoću nekog vanjskog utjecaja, okidača
  - Koristi se i kao memorijska ćelija



Izvor: Wikipedia

Integrirane verzije

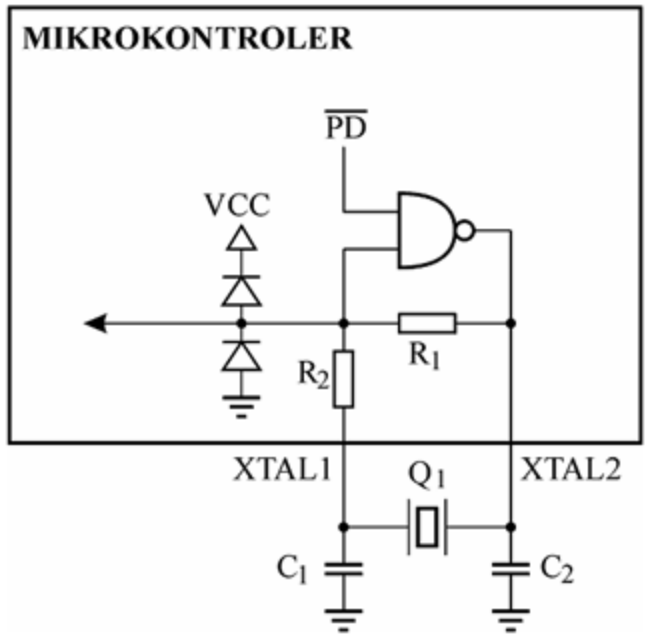
- Npr: NE 555 timer
- Tri načina rada
  - Astabilni
  - Monostabilni
  - Bistabilni

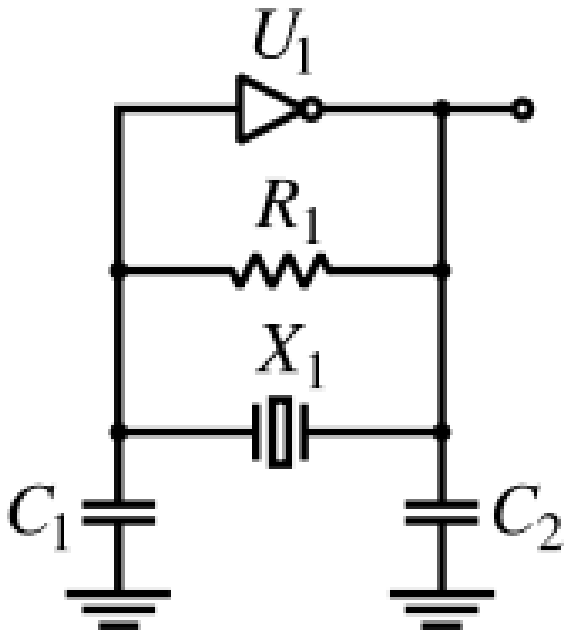


Oscilator s kristalom

- Pierceov oscilator
  - Kristal kao rezonantni sklop
- Podsjetimo se URS-a? ☺

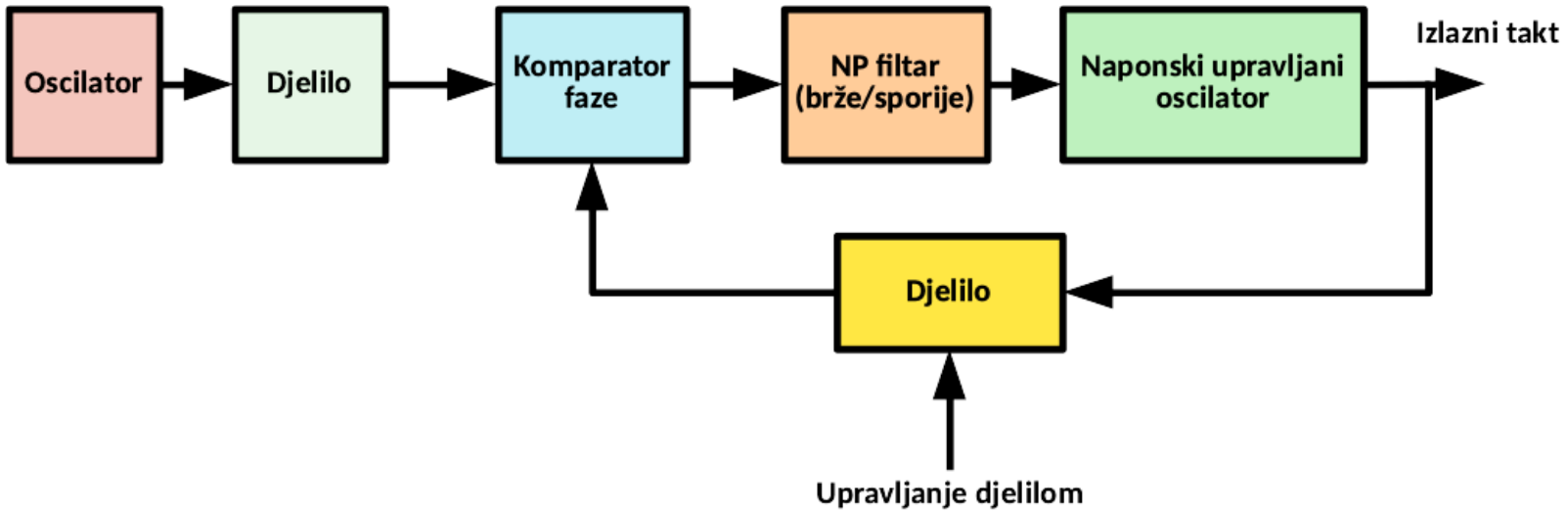
Oscilator mikrokontrolera 80C51





Stabilizirani elektronički oscilator

- Koristi rezonator (najčešće kristal) za stabilizaciju frekvencije
- PLL (Phase-lock loop) sklop
  - Sinkronizira vlastitu frekvenciju titranja s referentnom
  - Moguće izvesti množenje ili dijeljenje frekvencije
  - U osnovi isti princip i u atomskim oscilatorima



Usporedba oscilatora

- Zasnovani na mehaničkim rezonatorima
  - Kristali i keramički rezonatori
  - Visoka inicijalna preciznost
  - Umjereno mali temperaturni koeficijent
  - Osjetljivi na promjenu frekvencije (mehanička oštećenja)

- Zasnovani na elekričnim spojevima s faznim pomakom
  - RC – oscilator
  - Niska cijena
  - Preciznost ovisi o temperaturi i napajanju
  - Varijacije nominalne izlazne frekvencije 5% do 50%

Utjecaj okoline

- Okolina bitna za odabir oscilatora
  - Elektromagnetsko zračenje
  - Vibracije, udarci
  - Vlaga, temperatura
- Utjecaji
  - Treperenje faze signala
    - Kristali ispoljavaju male pomake frekvencije
  - Promjena izlazne frekvencije
    - Starenje
  - U težim slučajevima prekid rada
- Manji utjecaj okoline ako se koriste integrirane komponente

Usporedba oscilatora

Oscilator takta	Preciznost	Prednosti	Mane
Kristal	Srednja do velika	Niska cijena	Osjetljivost na EM zračenje, vibracije i vlagu
Kristal oscilator modul	Srednja do velika	Neosjetljivost na EM zračenje i vlagu, nepotrebne dodatne komponente	Visoka cijena, visoka potrošnja električne energije, osjetljivost na vibracije, velikih dimenzija
Keramički rezonator	Srednja	Niža cijena	Osjetljivost na EM zračenje, vibracije i vlagu
Ugrađeni silicijski oscilator	Mala do srednja	Neosjetljivost na EM zračenje i vlagu, nepotrebne dodatne komponente, male dimenzije	Temperaturna osjetljivost je generalno lošija nego kod kristala i kermamičkih oscilatora, visoka potrošnja električne energije
RC oscilator	Mala	Najniža cijena	Obično osjetljivi na EM zračenje, vlagu i temperaturu, velika potrošnja električne energije

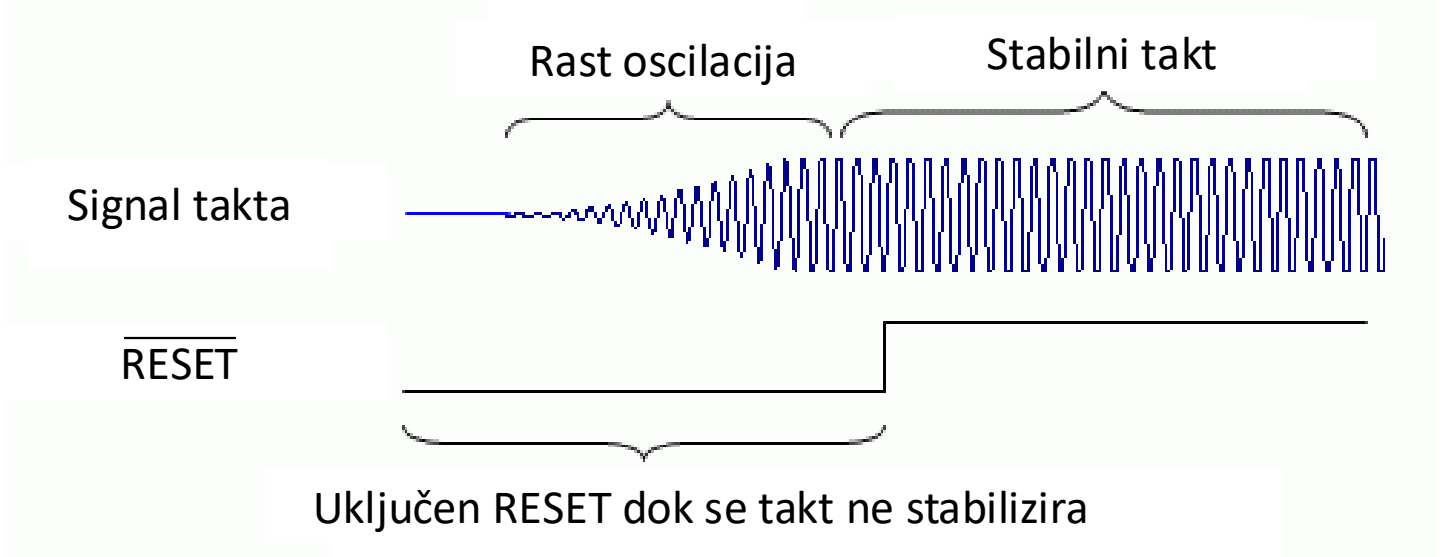
Preciznost

Oscilator takta	Točnost	Starenje (10 godina)	Potrebna snaga	Težina
Kristal (XO)	1E-5 do 1E-4	10-20 ppm	20 µW	20 g
Rubidij (Rb, RbXO)	1E-9	5E-10 do 5E-9	6-12 W	1,5 – 2,5kg
Cezij (Cs)	1E-12 do 1E-12	1E-12 do 1E-11	25-40 W	10 – 20 kg

- Kristalni oscilatori
  - Cijena ispod 5 US$
- Rubidijski i cezijski standardni oscilatori
  - Cijena veća od 40.000 US$

Pokretanje oscilatora

- Pokretanje oscilatora – uspostavljanje titranja rezonatora
  - Potrebna određena količina šuma koja zatitra rezonator (kristal) – šiljak napona pri uključivanju napajanja



Izvor: NEC Electronics

Primjer čekanja na uključivanje dok se ne stabilizira signal takta

- Buđenje iz SLEEP-a – nema početnog skoka napona

U praksi...

- Microchip PIC
  - Odabir više vrsta oscilatora, ovisno o uporabi sustava
    - Brzina
    - Potrošnja
    - Utrošak pinova
  - Vrste oscilatora:
    - HS
    - XT
    - LP
    - RC
    - IntRC
    - ER

- ESP32
  - Vanjski:
    - 40 Mhz kristalni oscilator
    - 10 ppm preciznost
  - RTC:
    - 32,768kHz kristal ili oscilator

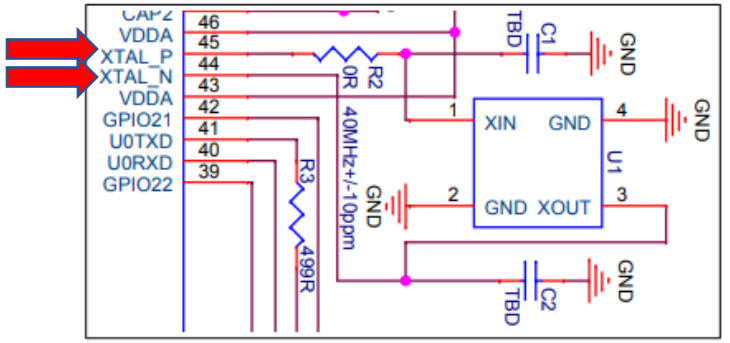


Figure 5: ESP32 Crystal Oscillator

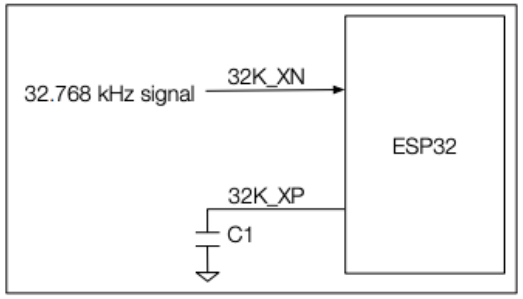


Figure 7: Schematic of External Oscillator

Interni RTC oscilator – greška *f* cca. 5% !!!

https://www.espressif.com/sites/default/files/documentation/esp32_hardware_design_guidelines_en.pdf

Vremenski sklopovi

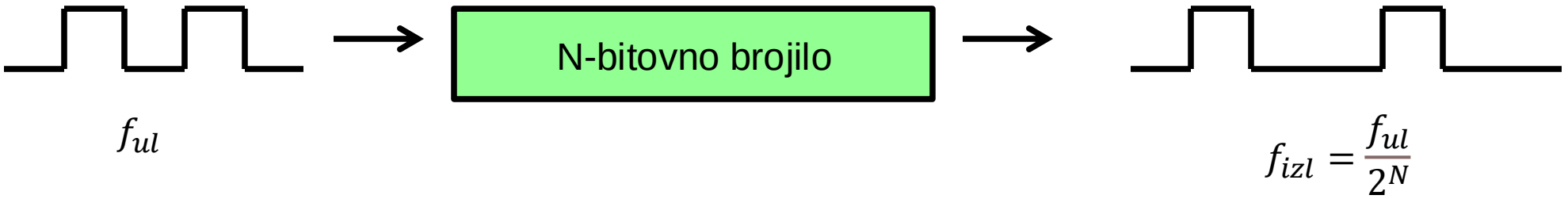
Vremenski sklopovi

- Sklopovi mikrokontrolera namijenjeni mjerenju vremena ili generiranju vremenski određenih signala

- Nekoliko primjera
  - Brojilo (*counter*)
  - Vremenski sklopovi (*timer*)
  - Sklopovi za zapis trenutnog vremena i usporedbu vremena (*capture & compare*)
  - Stvarno vrijeme (*real-time clock*)
  - *Pulsno-širinski modulator (pulse-width modulator)*
  - …

Brojilo

- U osnovi – slijedni logički sklop od N bistabila
- Broji u rasponu [0, 2^N-1] – prema gore ili dolje
- Ulaz
  - Pravokutni signal frekvencije f_{ul}
- Izlaz
  - Pravokutni signal frekvencije f_{ul} / 2^N

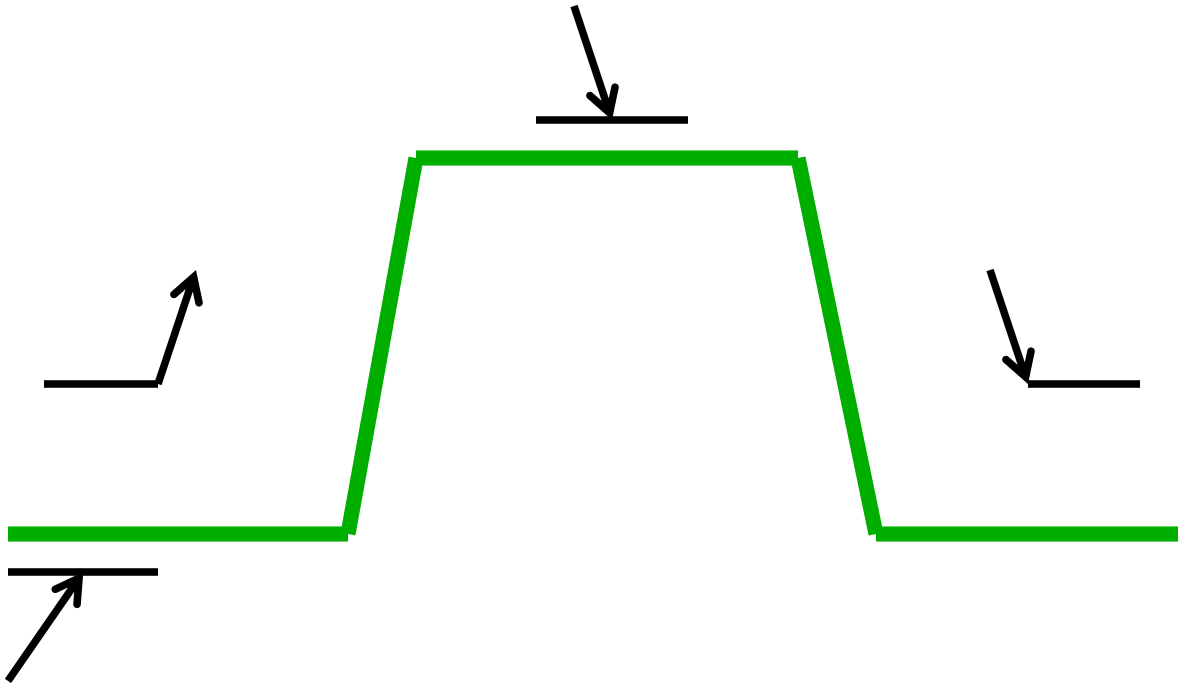


Brojilo

- Osnovna funkcija – brojenje
- Dodatni efekt – dijeljenje ulazne frekvencije
- Brojilo = dijelilo frekvencije
  - Želimo podijeliti frekvenciju s 5
  - Nemamo brojilo koje broji do 5
  - Rješenje
    - Uzmemo 3-bitno brojilo, broji u intervalu [0, 7]
    - U njega upišemo konstantu 4 i postavimo brojilo da broji prema dolje
      - 4, 3, 2, 1, 0
    - Kad dođe do 0 ponovno postavimo konstantu 4
  - Potrebno: brojilo, registar (konstanta) i nešto logike

Što brojiti?

- Pravokutni signal
  - Razina (visoka ili niska)
  - Uzlazni brid
  - Silazni brid
  - Svaki n-ti ulazni/silazni brid
  - …



Brojilo kao vremenski sklop

- Kontroliramo frekvenciju koju dovodimo na ulaz brojila
  - Poznata frekvencija = poznato trajanje perioda
  - Imamo sklop kojim možemo mjeriti vrijeme
- Osnova vremenskih sklopova (*timer*)
- Dodatna kontrolna logika
  - Generiranje prekida na istek vremena (brojilo = 0)
  - Upravljanje radom

Dodatne mogućnosti

- Upravljanje izvorom ulaznog signala u brojilo
  - Unutarnji izvor – takt
  - Vanjski izvor – proizvoljni izvor impulsa (oscilator, senzor)
  - Ulančavanje više brojila (povećanje raspona)
  - Odabir pred-dijelila za smanjenje frekvencije

- Upravljanje *okidanjem* brojila
  - Brojenje razine, brida, ...

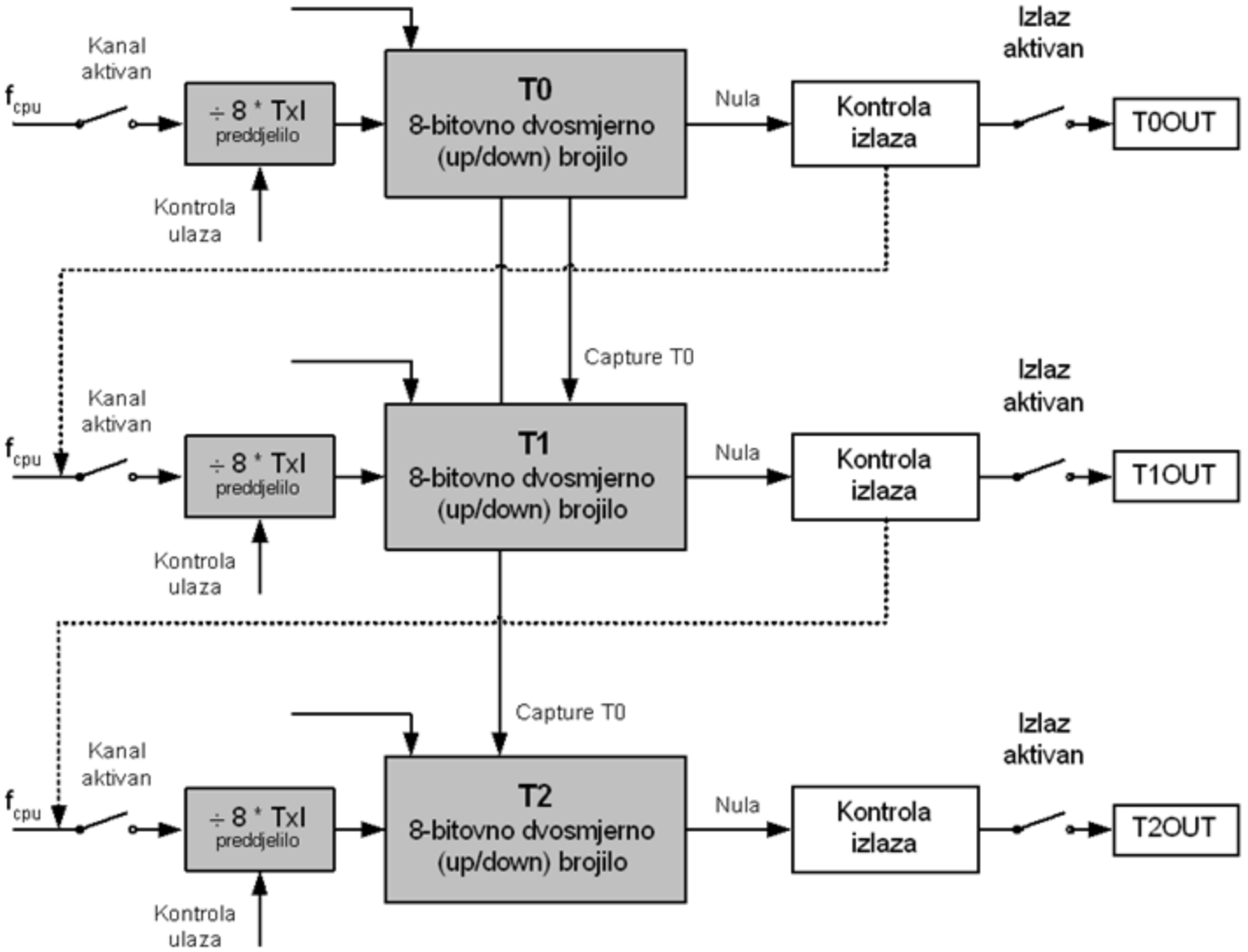
- Upravljanje radom
  - Zaporni sklop (*gated-timer*)
    - Moguća kontrola rada (izvana i iznutra)
    - Ulaz koji određuje radi li *timer* ili ne
      - Štoperica (stani-kreni)
  - Određivanje smjera brojenja (*up-down*)
    - Upravljanje smjerom brojenja (izvana i iznutra)

Capture sklop

- Upravljanje pohranom
  - Pohrana *snimke* trenutnog stanja brojila (*capture*)
  - Trenutak *okidanja* određen vanjskim signalom
  - Okidanje – trenutno stanje brojila pohranjuje se u poseban registar
- "Štoperica" s *prolaznim vremenom*

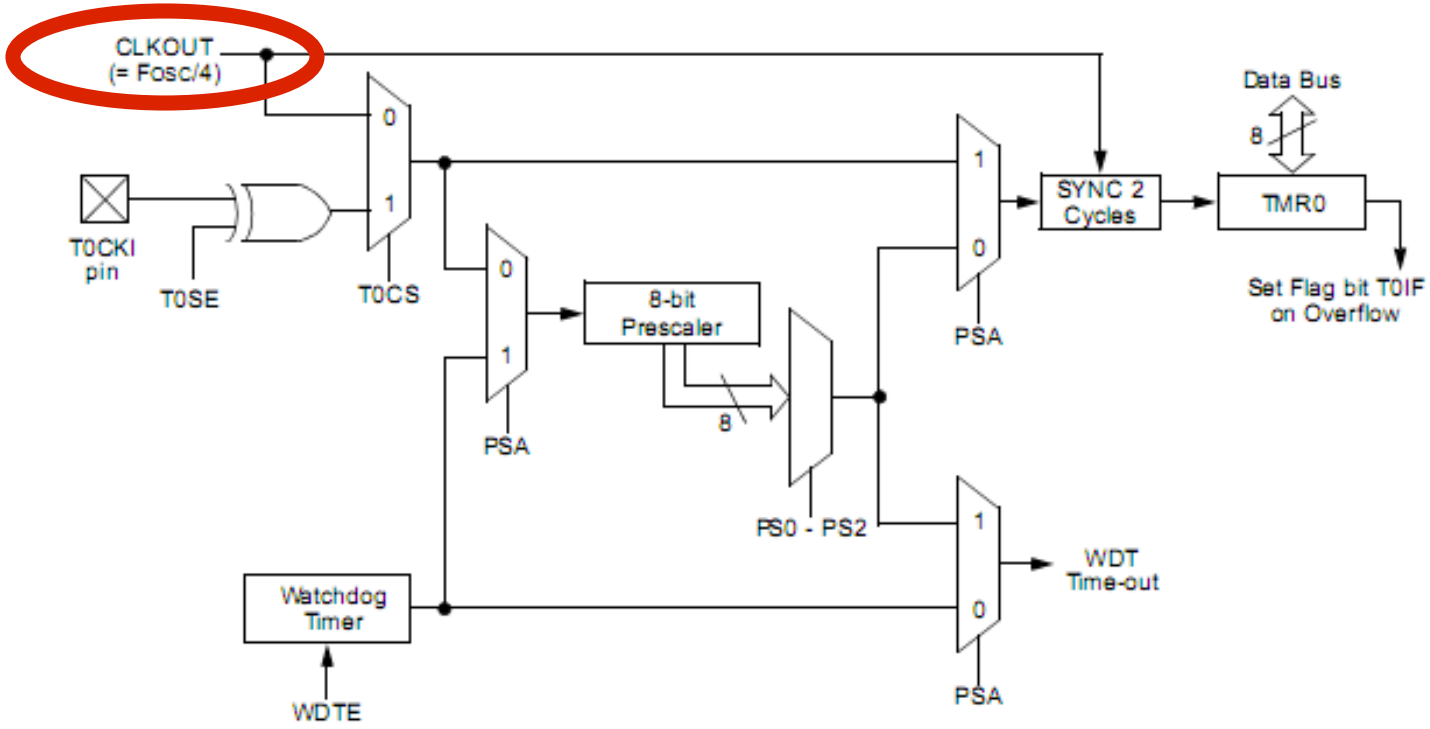
Primjeri vremenskih sklopova

- FRISC ?! ☺



Primjeri vremenskih sklopova

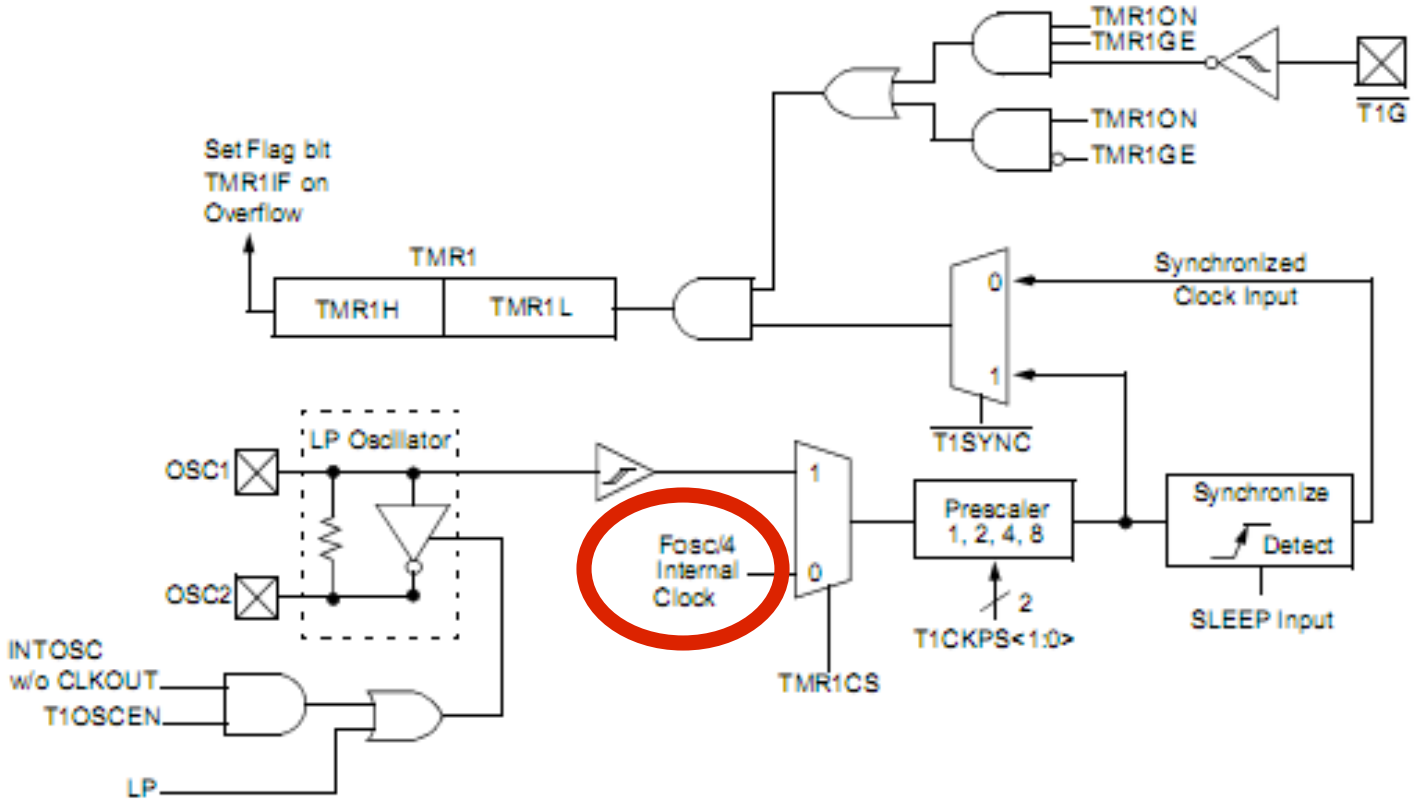
- Primjer: PIC12 - Timer0
  - 8-bitno brojilo + preddjelilo (*prescaler*)
  - Unutarnji/vanjski takt, odabir brida
  - Prekid na prijelazu FF → 00



Note 1: T0SE, T0CS, PSA, PS0-PS2 are bits in the Option register.

Primjeri vremenskih sklopova

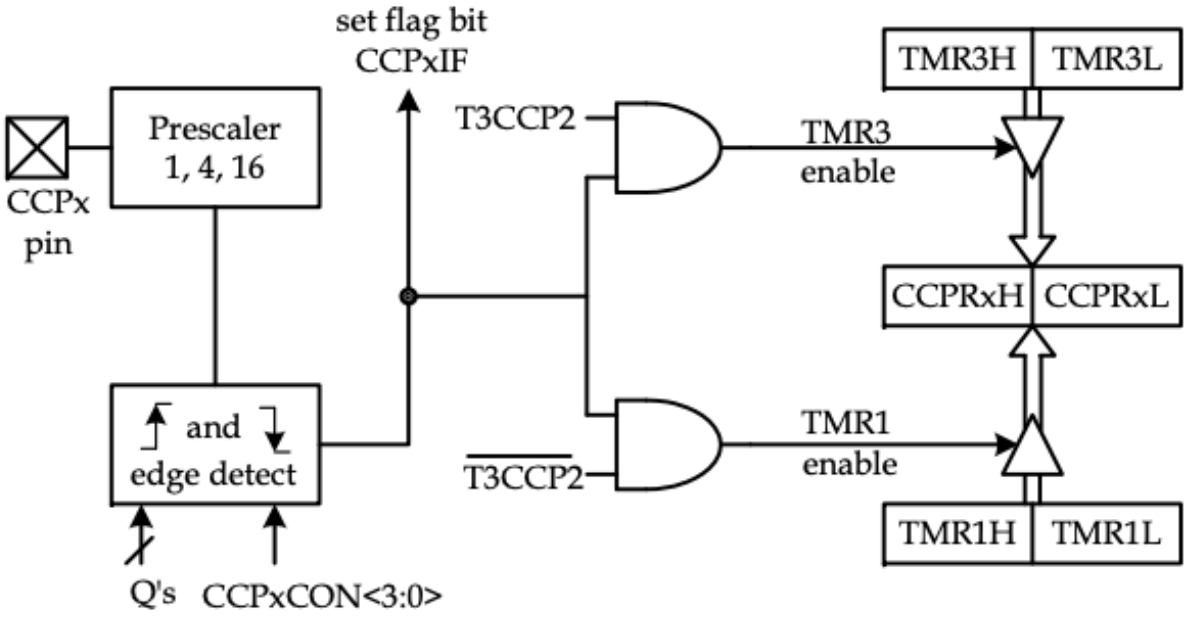
- Primjer: PIC12 - Timer1
  - 16-bitno brojilo + preddjelilo (*prescaler*)
  - Unutarnji ili vanjski takt
  - Zaporni sklop (*gated timer*)
  - Prekid na prijelazu FFFF → 0000



Primjeri vremenskih sklopova

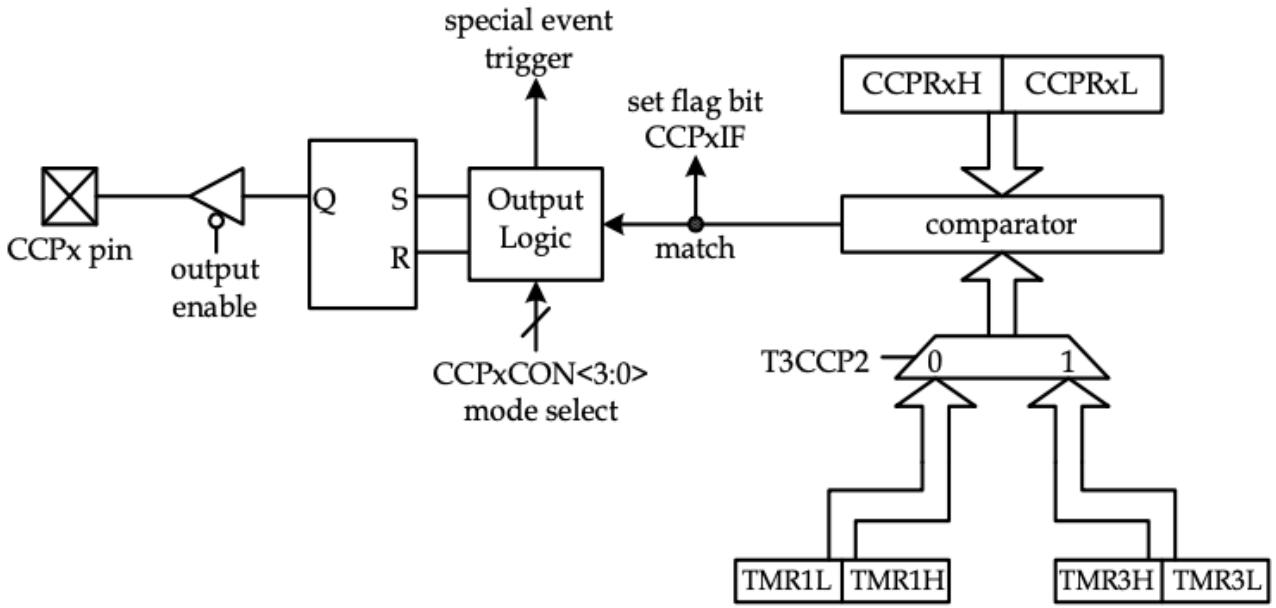
- PIC18 Capture
  - Zapis stanja vremenskog sklopa u nailasku *događaja*
  - *događaj -> brid ulaznog signala*
  - *Svaki ili svaki n-ti brid*
  - *Podizanje prekida na događaj*

- Zašto *capture* sklop???
  - Zašto ne kopirati stanje vremenskog sklopa programski?



Primjeri vremenskih sklopova

- PIC18 Compare
  - Stanje vremenskog sklopa se uspoređuje sa stanjem registara
  - *U slučaju podudaranja -> događaj*
  - *Događaj se može reflektirati na stanje izlaznog pina*
    - *prelazi u visoko stanje*
    - *prelazi u nisko stanje*
    - *prebacuje stanje (toggle)*
    - *ostaje nepromijenjeno stanje*
  - *Događaj može resetirati vremenski sklop*



- Zašto *compare* sklop???
  - Zašto ne usporedba stanja vremenskog sklopa programski?

RTC

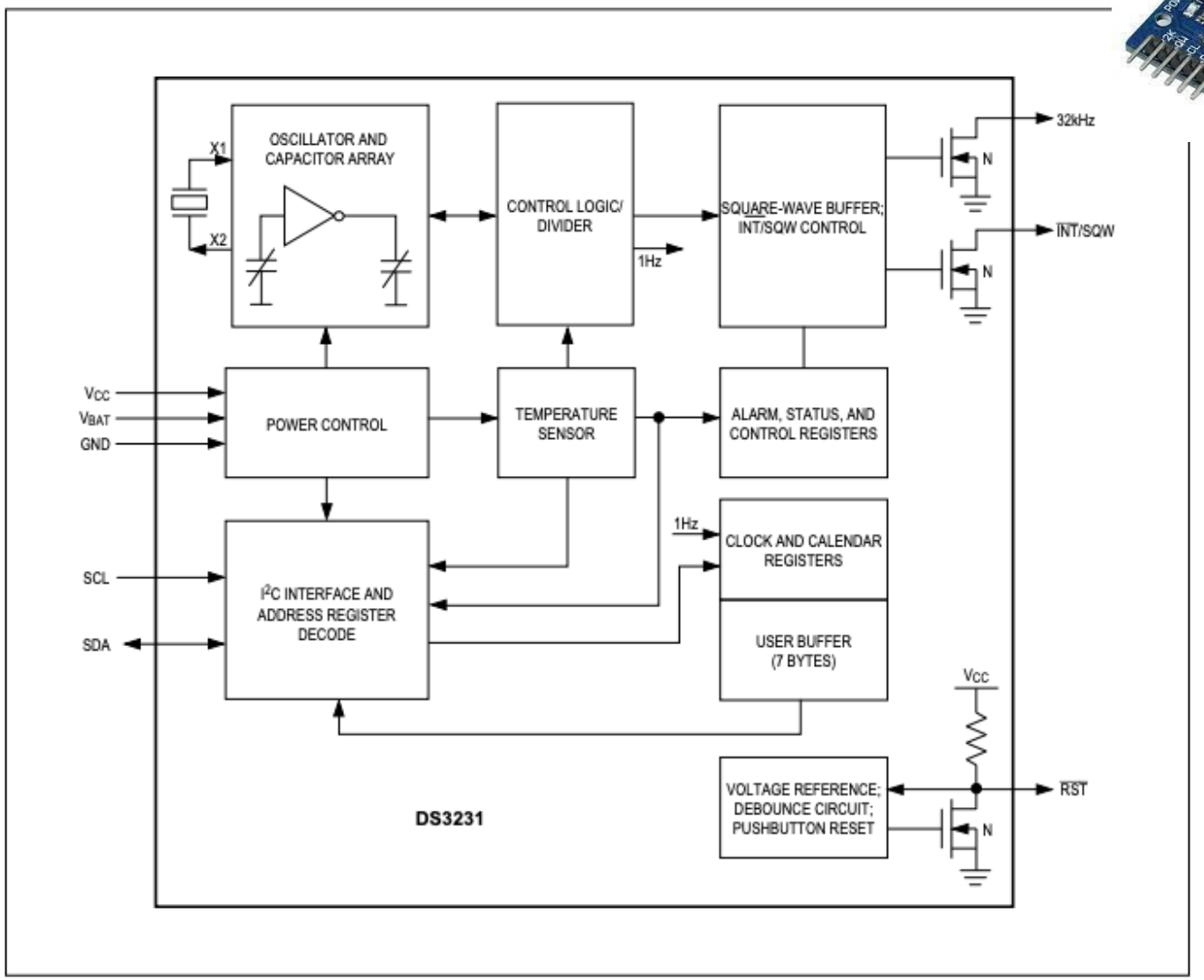
- S vremenskim sklopovima možemo mjeriti protok vremena – relativno vrijeme

- Za informaciju o apsolutom vremenu – real-time clock
  - minuta, sat, dan, datum, alarmi
  - poželjno zadržati funkcionalnost i tijekom prestanka napajanja sustava i/ili stanja niske potrošnje (*deep sleep*)

- Vrste RTC-a
  - interni – ugrađen u mikrokontroler
  - eksterni – zasebni IC povezani sabirnicom s mikrokontrolerom

Primjer – DS3231

- Održava:
  - Sekunde
  - Minute
  - Sate
  - Dan
  - Datum
  - Mjesec
  - Godina
  - Korekcije broja dana u mjesecu, prestupna godina …
- Dva alarma
- Baterijsko *backup* napajanje
- I²C sučelje
- Temperaturno kompenziran
- Točnost: 2ppm 0 - 40 C

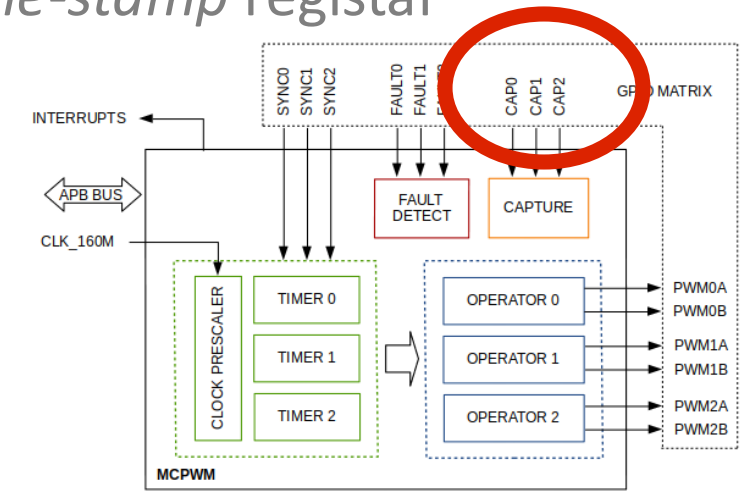


ESP32 vremenski sklopovi

- Dvije grupe vremenskih sklopova
  - Svaka grupa dva vremenska sklopa
- Svaki vremenski sklop
  - 64-bitni up/down brojač
  - 16-bitni prescaler
  - Izvor vremenskog signala: APB_CLK – 80MHz
  - *Alarm (očekivana vrijednost timera):*
    - *Prekid (opcionalno)*
    - *Auto reload (opcionalno)*

https://docs.espressif.com/projects/esp-idf/en/v4.3/esp32/api-reference/peripherals/timer.html

- Capture:
  - MCPWM modul
  - 32-bitni timer, APB_CLK (80MHz)
  - Capture prescale (dijeljenje ulaznih bridova)
  - *Time-stamp* registar



MCPWM Block Diagram

https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/peripherals/mcpwm.html#capture

ESP RTC

- RTC čuva vrijeme sustava i tijekom reseta i modova spavanja
  - Power-up reset također resetira i RTC (nema pomoćnog baterijskog napajanja)
  - Preciznost ovisi o izvoru signala vremenskog vođenja
    - Interni 150 kHz RC oscilator (najmanja potrošnja, loša stabilnost frekvencije, nema vanjskih komponenti)
    - Eksterni 32 kHz kristal (dobra stabilnost frekvencije, 1uA veća potrošnja u modu niske potrošnje, jedna vanjska komponenta)
    - Eksterni 32 kHz oscilator (gotov vanjski signal, dodatno vanjsko sklopovlje)
    - Interni 8,5 MHz oscilator (cca. 33 kHz nakon djelila, dobra frekvencijska stabilnost, 5uA veća potrošnja u modu niske potrošnje, nema vanjskih komponenti)
- SNTP za dobavu vremena nakon uključenja i periodičku korekciju drifta RTC-a
  - Spajanje na NTP poslužitelj korištenjem WiFi sučelja

Rad u stvarnom vremenu

Definicije sustava za rad u stvarnom vremenu

- Oxford Dictionary of Computing:

A Real Time System - "Any system in which the **time at which output is produced is significant**. This is usually because the input corresponds to some movement in the physical world, and the output has to relate to that same movement. The lag **from input time to output time must be sufficiently small** for acceptable timeliness."

- Young S. (1982), Real Time Languages

any information processing activity or system which has to **respond** to externally generated input stimuli **within a finite and specified period.**

- The Predictably Dependable Computer Systems (PDCS)

A real-time system is a system that is required to react to stimuli from the environment (including the passage of physical time) **within time intervals dictated by the environment**

Rad u stvarnom vremenu

- Koliko je sati? KADA će biti ručak?
- ZA KOLIKO će biti gotov ručak?
- Odziv (vrijeme od ulaza do izlaza)
- Relativnost, ovisno o okolini
  - krstareća raketa Tomahawk, proizvodnja automobila i UNIX (višekorisnički rad) nemaju iste zahtjeve
- Predvidljivost, KADA?
- Ispravnost rada računala ne samo rezultat izračunavanja već i vrijeme kada (za koliko) je rezultat proizveden!

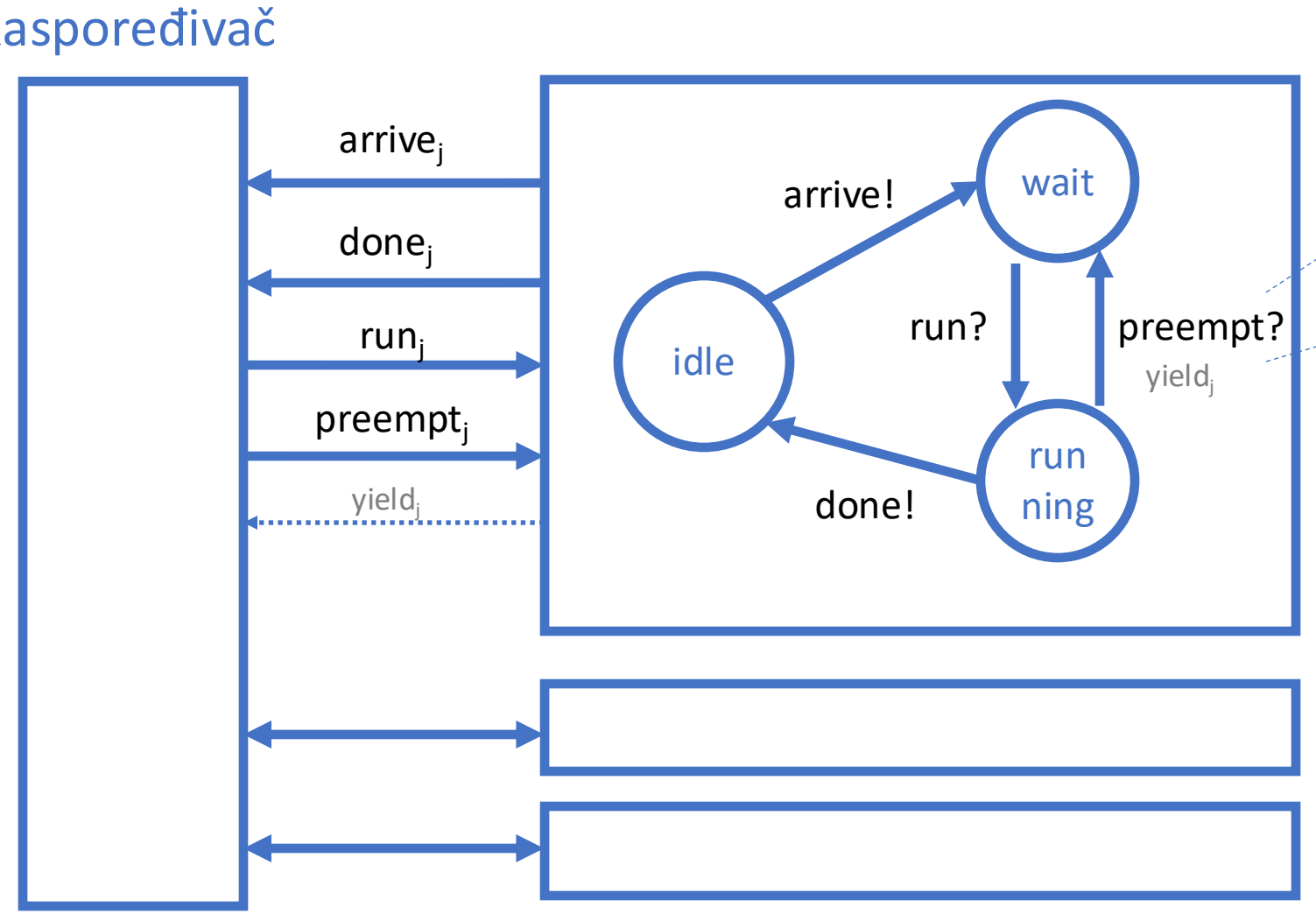
Zadaci u stvarnom vremenu

- Strogost poštivanja vremenskih ograničenja
  - Strogi sustavi (hard real-time systems) [aktiviranje zračnog jastuka]
  - Ublaženi sustavi (soft real-time systems) [dekodiranje video sadržaja]

- Poslovi s obzirom na vrijeme dolaska:
  - Periodički
  - Aperiodički

- Raspoređivač poslova:
  - Optimalno rasporediti poslove po dostupnim resursima i s obzirom na ovisnosti između poslova, tako da sva vremenska ograničenja obavljanja poslova budu poštovana
  - Osnovni zahtjevi na raspoređivač poslova:
    - U jednom trenutku samo jedan posao aktivan (po jezgri)
    - Svaki posao mora dobiti „dovoljno” procesorskog vremena za njegovo obavljanje u zadanom roku

Raspoređivač i poslovi



Multitasking:

- cooperative
- preemptive
- non-preemptive

[embeddedOS]
(Windows < 95)
macOS
TinyOS
JavaScript ?!?!

Model periodičkog posla

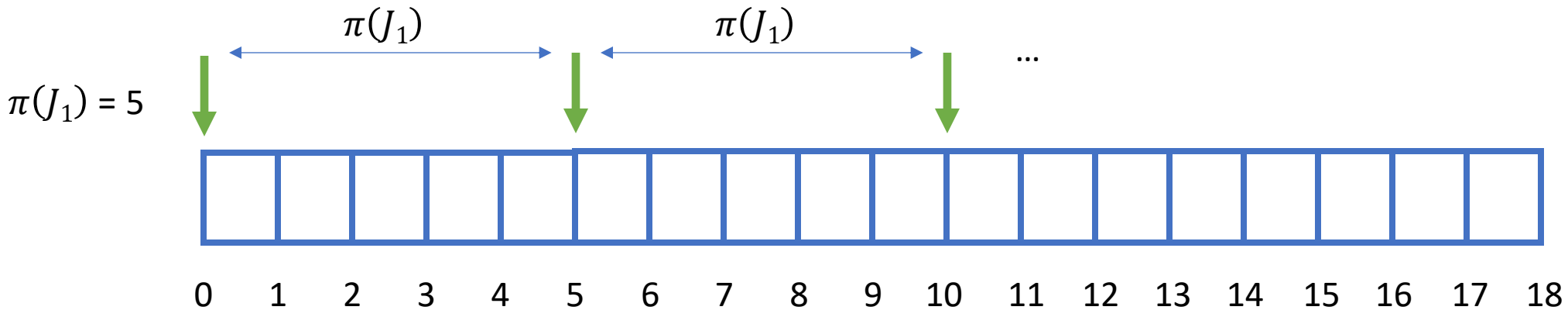
- Vrijeme razdijeljeno u vremenske odsječke t_0, t_1, t_2, … uniformnog trajanja

- Periodički posao opisan s tri ključna parametra:
  - $\pi(J)$ - Period dolaska posla
  - $\delta(J)$ - Krajnji rok dovršetka posla
  - $\eta(J)$ - Najdulje vrijeme izvršavanja posla (*worst-case execution time*)

- Parametri $\pi(J)$ i $\delta(J)$ su definirani tijekom oblikovanja sustava
- Parametar $\eta(J)$ je posljedica implementacije sustava
  - Karakteristike odabrane izvršne platforme
  - Implementacija programske podrške

Period dolaska posla

- Posao J se izvršava periodički, svakih $\pi(J)$ vremenskih odsječaka
- Dolazak instance posla J^a
  - J^a je a-ta instanca posla J, $a \in \mathbb{N}$
  - *Vrijeme dolaska (arrival time)* instance posla: (a-1) $\pi(J)$
  - Instanca posla J^a je inicijalno u stanju *idle*
  - Raspoređivač šalje događaj $arrive_j$, instanca posla prelazi u stanje *wait* i čeka da joj se dodijeli vrijeme na procesoru/jezgri



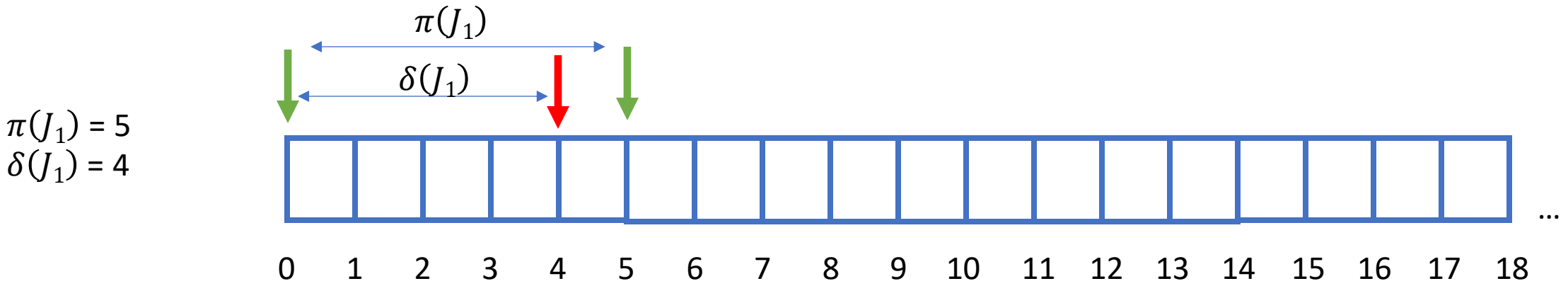
Krajnji rok dovršetka posla

- Dospjeli posao J mora biti završen za $\delta(J)$ vremenskih odsječaka, s obzirom na vremenski odsječak dolaska
- S obzirom na događaje, vrijedi

$$done_j - arrivej \leq \delta(J)$$

- Također mora vrijediti

$$\delta(J) \leq \pi(J)$$



Najdulje vrijeme izvršavanja posla

- Ocjenjuje se najgori slučaj trajanja izvršavanja posla J (izražen u vremenskim odsječcima)
  - Statička analiza
  - Analiza tijekom izvršavanja (dinamička analiza)

- Posao J garantira izdavanje događaja $done_j$ ako provede $\eta(J)$ vremenskih odsječaka u stanju *running*

- Da bi posao J bio rasporediv, mora vrijediti

$$\eta(J) \leq \delta(J) \leq \pi(J)$$

Npr. posao J je rasporediv s danim parametrima:

$\pi(J)$=5, $\delta(J)$=4, $\eta(J)$=3

Statička ocjena vremena izvršavanja posla

- Posao:
  - Slijed izraza (sastavljen od atomičnih izraza)
  - Ne postoje petlje (for …)
  - Postoje uvjetni izrazi (if …)

- Nizovi izraza:

$$\eta(stmt) = \eta(\text{stmt}_1) + \eta(\text{stmt}_2) + \ldots + \eta(\text{stmt}_i)$$

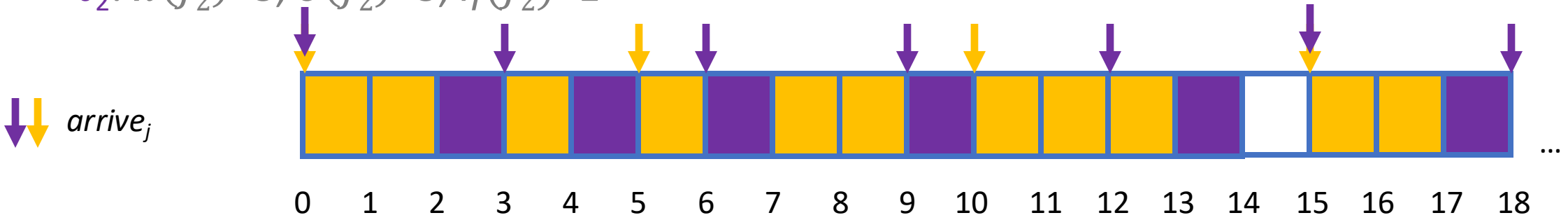
- Uvjetni izrazi:

$$\eta(stmt) = \eta(\text{e}) + \max\{\eta(\text{stmt}_1), \eta(\text{stmt}_2)\}$$

- Trajanje izvođenja izraza stmt_n, npr. $c = a + b$ ovisi o:
  - generiranom programskom kodu (lokalne/globalne varijable …)
  - arhitekturi (registarsko polje -> priručna memorija -> memorija …)
  - trenutnom stanju registara, priručne memorije …
  - …
- Problem suviše pesimističnih procjena trajanja izvršavanja
  - Npr. promašaj u priručnoj memoriji se dešava najčešće samo na početku bloka instrukcija koji koristi neki skup varijabli
  - Što s pogreškama u predviđanju grananja (i posljedičnim pražnjenjem cjevovoda …)?

Model periodičkih poslova

- Konačan skup periodičkih poslova $\Im$, gdje:
  - svaki posao J ima pridruženi period $\pi(J)$, krajnje vrijeme izvršavanja $\delta(J)$ i najdulje vrijeme izvršavanja $\eta(J)$
  - Za svaki posao J vrijedi $\eta(J) \leq \delta(J) \leq \pi(J)$
- Zadatak raspoređivača je pronaći raspored poslova u raspoložive vremenske odsječke (procesorsko vrijeme) tako da su zadovoljeni krajnji rokovi izvršavanja za sve poslove
- Npr. neka su dani poslovi J_1 i J_2 sa sljedećim parametrima:
  - J_1: $\pi(J_1)$=5, $\delta(J_1)$=4, $\eta(J_1)$=3
  - J_2: $\pi(J_2)$=3, $\delta(J_2)$=3, $\eta(J_2)$=1



Rasporedivost poslova

- Zadatak raspoređivača: pronalazak rasporeda izvršavanja (periodičkih) poslova, tako da svi poslovi budu obavljeni prije isteka njihovih krajnjih vremena izvršavanja
- Formalno: raspored σ za periodički model poslova $\mathfrak{J}$ je definiran kao funkcija:

$$\sigma : \mathbb{N} \rightarrow \mathfrak{J} \cup \{\perp\}$$

$\sigma(t) = J$: u vremenskom odsječku t izvodi se posao J

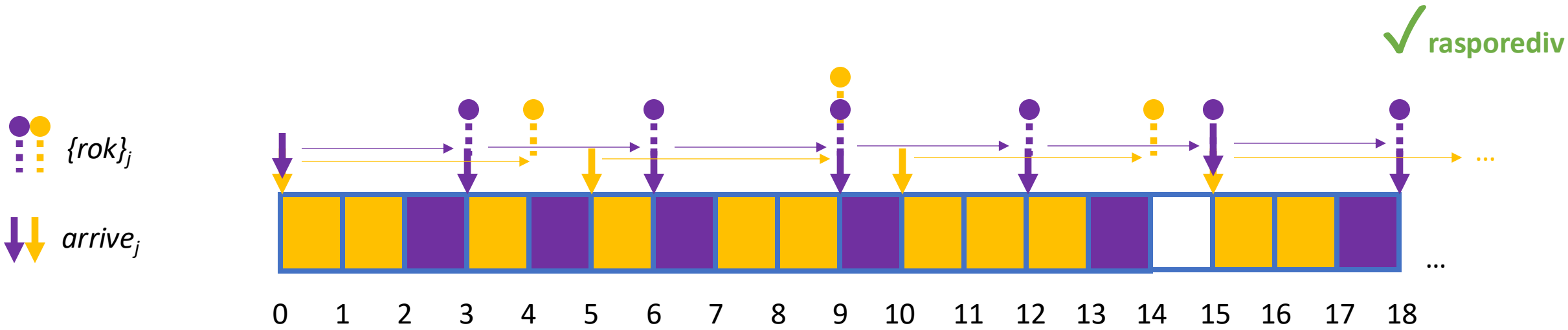
$\sigma(t) = \perp$: u vremenskom odsječku t ne izvodi se niti jedan posao

- raspored σ je sukladan rokovima (*deadline-compliant*) za periodički posao J ako alocira dovoljan broj vremenskih odsječaka $\eta(J)$ tako da posao J završi prije roka $\delta(J)$
- raspored σ je **sukladan rokovima za periodički model poslova $\mathfrak{J}$** ako je sukladan rokovima za svaki od poslova iz modela poslova $\mathfrak{J}$

Rasporedivost poslova

- Npr. neka su dani poslovi J_1 i J_2 sa sljedećim parametrima:
  - J_1: $\pi(J_1)$=5, $\delta(J_1)$=4, $\eta(J_1)$=3
  - J_2: $\pi(J_2)$=3, $\delta(J_2)$=3, $\eta(J_2)$=1

- Svi poslovi završeni prije krajnjih rokova -> postoji σ koji je sukladan rokovima za $\Im$ -> model poslova $\Im$ je rasporediv

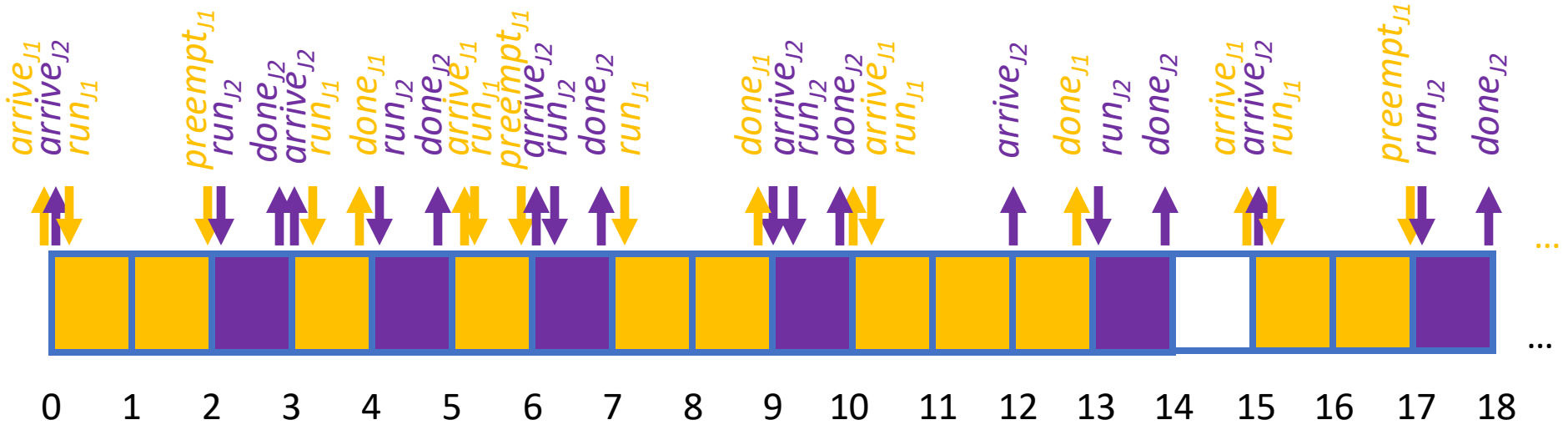


Interakcija raspoređivača i poslova

Npr. neka su dani poslovi J_1 i J_2 sa sljedećim parametrima:

J_1: $\pi(J_1)$=5, $\delta(J_1)$=4, $\eta(J_1)$=3

J_2: $\pi(J_2)$=3, $\delta(J_2)$=3, $\eta(J_2)$=1



Periodički raspored poslova

- σ je periodički raspored s periodom *p*, gdje je p ∈ ℕ ako za svaki *t* vrijedi

$$t \geq 0: \ \sigma(t+p) = \sigma(t)$$

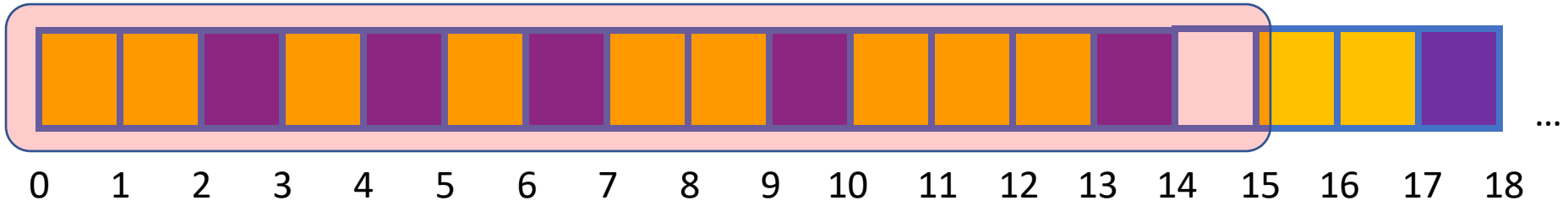
- Periodički model poslova $\mathfrak{J}$ je rasporediv ako i samo ako postoji periodički raspored sukladan rokovima za model poslova $\mathfrak{J}$

- Period *p* je najmanji zajednički višekratnik brojeva u skupu $\{\pi(J) \mid J \in \mathfrak{J}\}$

J_1: $\pi(J_1)$=5, $\delta(J_1)$=4, $\eta(J_1)$=3
J_2: $\pi(J_2)$=3, $\delta(J_2)$=3, $\eta(J_2)$=1

p=15

✓ rasporediv



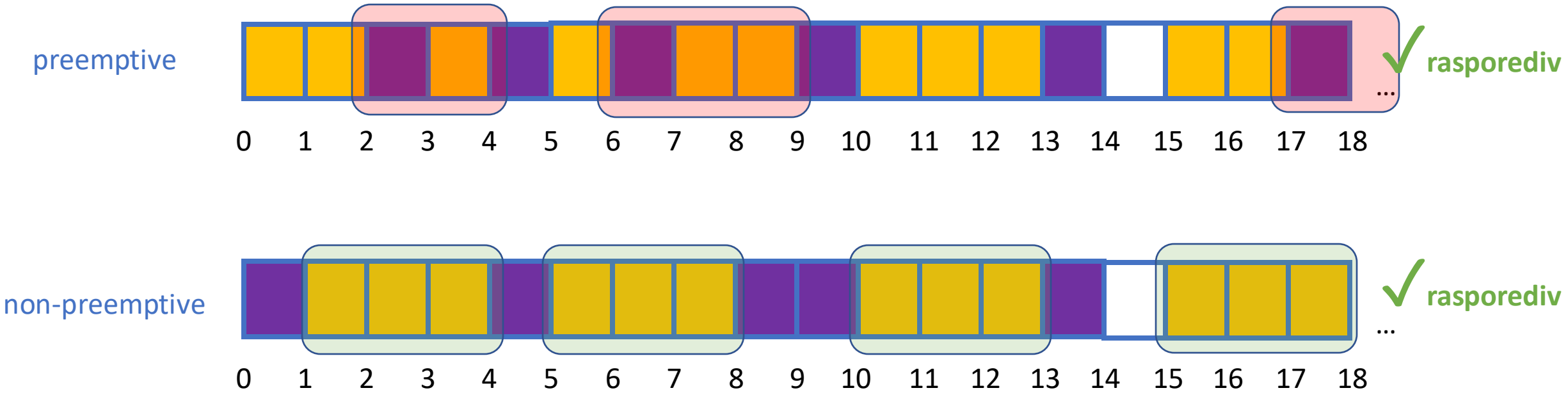
Iskorištenje

- Iskorištenje (*utilization*) periodičnog modela poslova $\mathfrak{J}$
- Koliko procesnog vremena je potrebno da bi bila moguća obrada svih poslova

$$U(\mathfrak{J}) = \sum_{J \in \mathfrak{J}} \eta(J) / \pi(J)$$

- Brzi test rasporedivosti: $U(\mathfrak{J}) > 1$ podrazumijeva da nedostaje procesnog vremena za poslove iz $\mathfrak{J}$, stoga model poslova nije rasporediv

Preemptive raspoređivanje



- Preemptive raspoređivač – može prekinuti izvođenje posla i naknadno ga nastaviti
  - Fleksibilnost u raspoređivanju izvršavanja poslova
  - Lakše pronaći raspored sukladan rokovima modela posla ukoliko se koristi *preemptive* raspoređivanje (npr. periodični model može biti rasporediv samo uz korištenje *preemptive* raspoređivanja)
  - Nedostatak *preemptive* raspoređivača: gubitci procesorskog vremena tijekom zamjene konteksta poslova (želimo što manje zamjena)!

Politike raspoređivanja

- Politika raspoređivanja, uz dani periodični model poslova $\mathfrak{I}$ treba:
  - Odrediti periodički raspored σ sukladan rokovima ili
  - Prijaviti nemogućnost nalaženja takvog rasporeda
- Velik broj politika (algoritama) raspoređivanja
  - Osnovna podjela: fiksni ili dinamički prioritet poslova
- Neke od svojstava politika:
  - Računalna složenost
    - Linearna, polinomijalna, NP složenost s obzirom na broj poslova u $\mathfrak{I}$ i period p
  - On-line ili off-line raspoređivanje
    - Prethodno izračunati raspored, računanje rasporeda prije svakog vremenskog odsječka, računanje na promjenu u skupu $\mathfrak{I}$ …
  - Alternativni kriteriji raspoređivanja
    - Ako postoji više rasporeda koji zadovoljavaju kriterij sukladnosti rokovima, kako odabrati *onaj pravi?*
    - Minimizacija broja *preemption* operacija, maksimizacija responzivnosti …

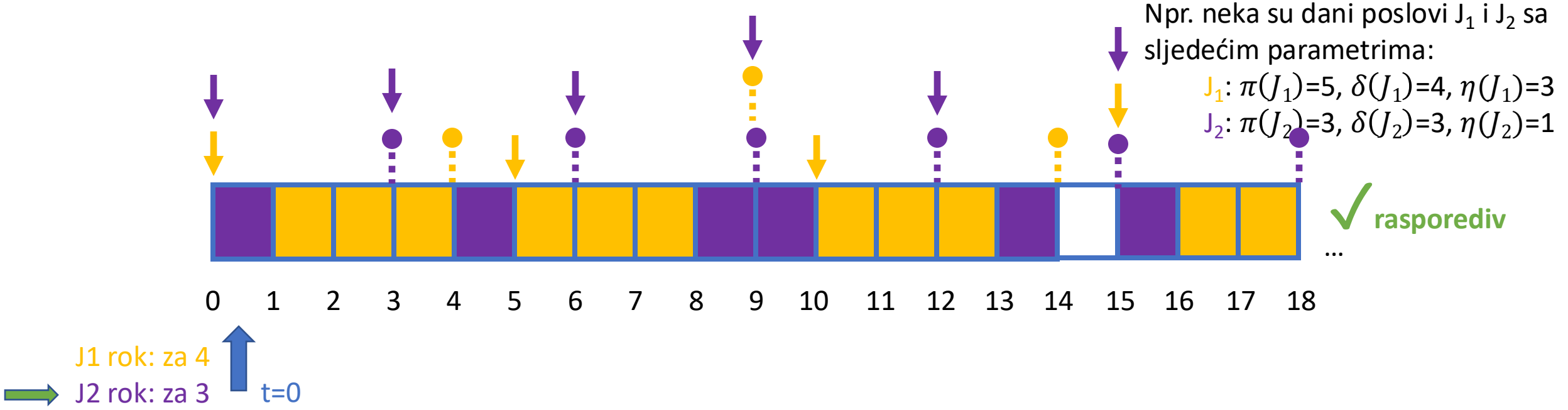
Drugi modeli poslova

- Ovisnosti između poslova
  - npr. posao J_4 ovisi o prethodnom završetku posla J_1
- Dinamički promjenljivi skup poslova
  - dodavanjem ili brisanjem posla mora se obaviti test rasporedivosti, stvoriti novi raspored bez utjecaja na izvršavanje sustava i dostizanje krajnjih rokova poslova
- Aperiodični poslovi
  - Nepoznat trenutak dolaska posla
  - Rezervacija vremena u rasporedu za aperiodičke poslove
  - „Best-effort" umjesto garantiranja dostizanja rokova

- Višeprocesorske/višejezgrene platforme
  - Raspoređivanje poslova i po više procesora ili jezgri
  - Afiniteti poslova za jezgre
  - Specijalizirani procesori / jezgre
- Hard / soft zahtjevi na dostizanje krajnjih rokova
  - Hard – garancija rasporeda za obavljanje svih poslova na vrijeme (npr. abs)
  - Soft – garancija rasporeda za određeni % obavljanja poslova na vrijeme (npr. dekodiranje videa)

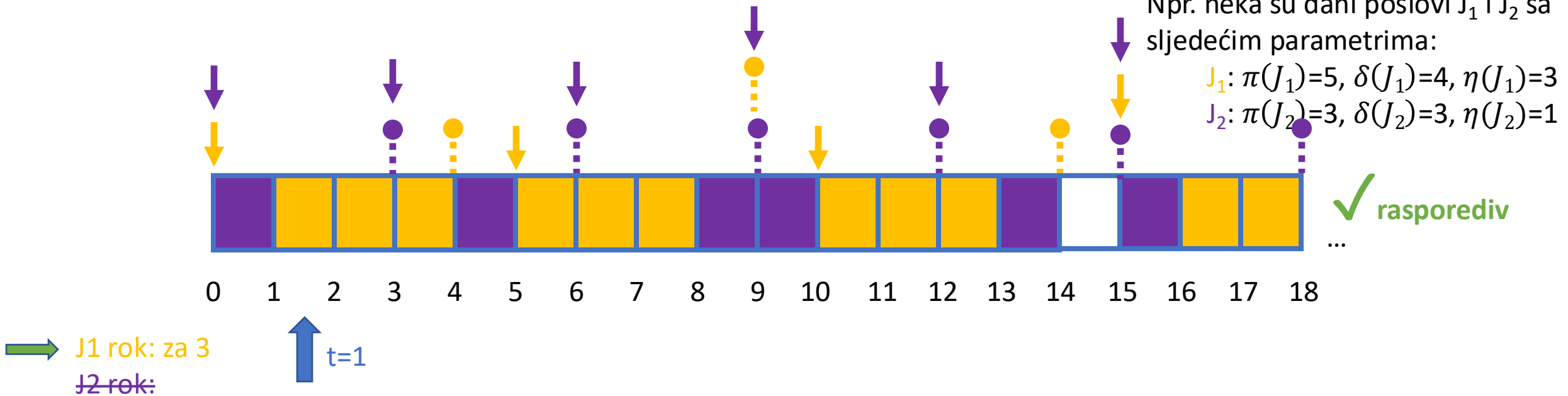
Politika najbližeg roka

- *Eariest deadline first* (EDF) politika:
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(t) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je rok $\delta(J_e)$ najbliži t: $\sigma(t) = J_e$
    - *Ako ima više poslova s istim najbližim rokom, odabir se vrši arbitrarno ili po alternativnom kriteriju*



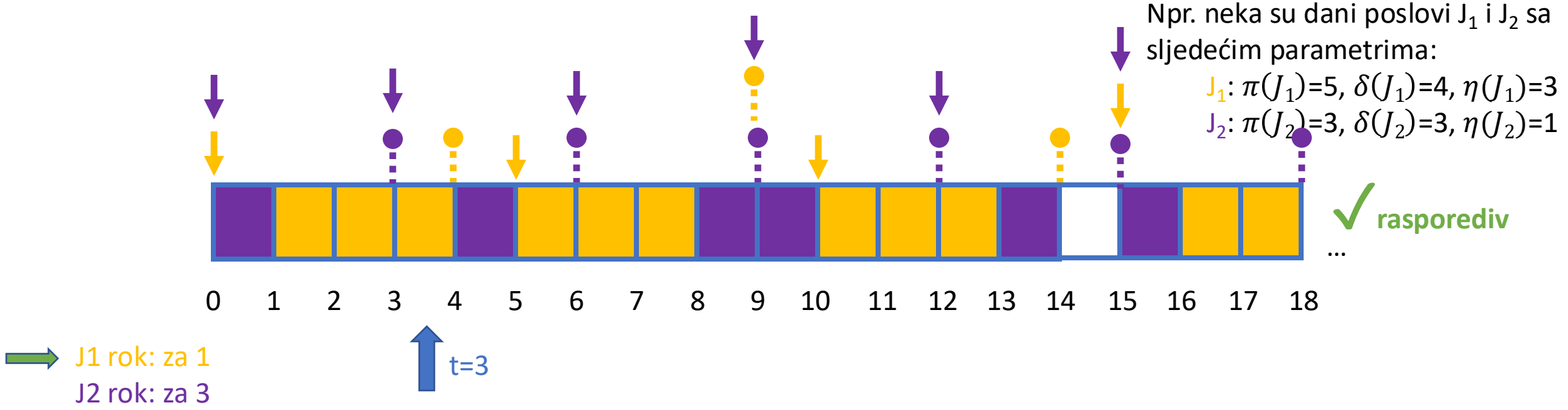
Politika najbližeg roka

- *Eariest deadline first* (EDF) politika:
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(t) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je rok $\delta(J_e)$ najbliži t: $\sigma(t) = J_e$
    - *Ako ima više poslova s istim najbližim rokom, odabir se vrši arbitrarno ili po alternativnom kriteriju*



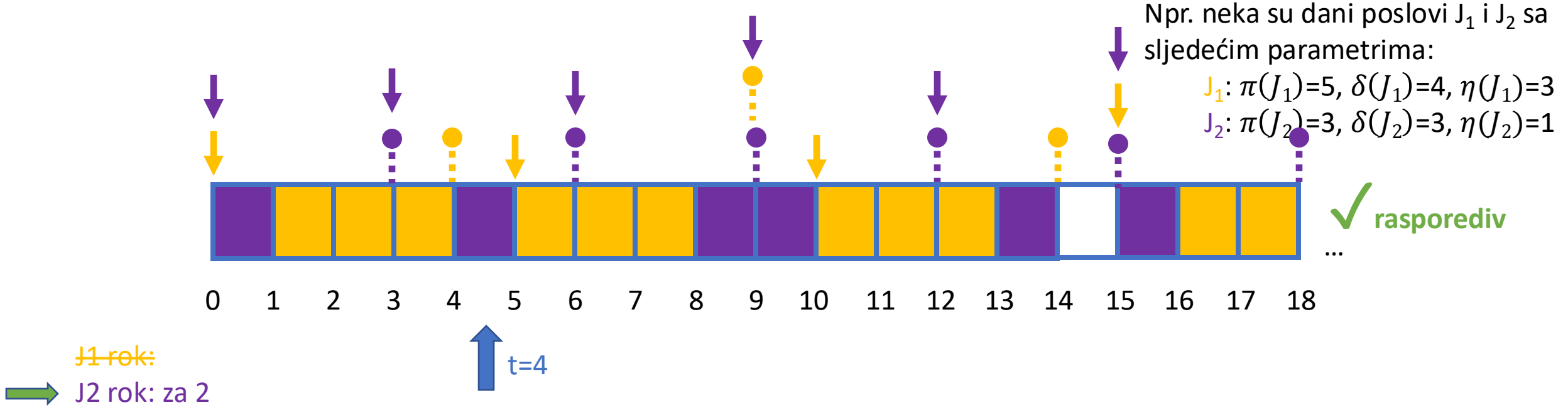
Politika najbližeg roka

- *Eariest deadline first* (EDF) politika:
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(t) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je rok $\delta(J_e)$ najbliži t: $\sigma(t) = J_e$
    - *Ako ima više poslova s istim najbližim rokom, odabir se vrši arbitrarno ili po alternativnom kriteriju*



Politika najbližeg roka

- *Eariest deadline first* (EDF) politika:
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(t) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je rok $\delta(J_e)$ najbliži t: $\sigma(t) = J_e$
    - *Ako ima više poslova s istim najbližim rokom, odabir se vrši arbitrarno ili po alternativnom kriteriju*



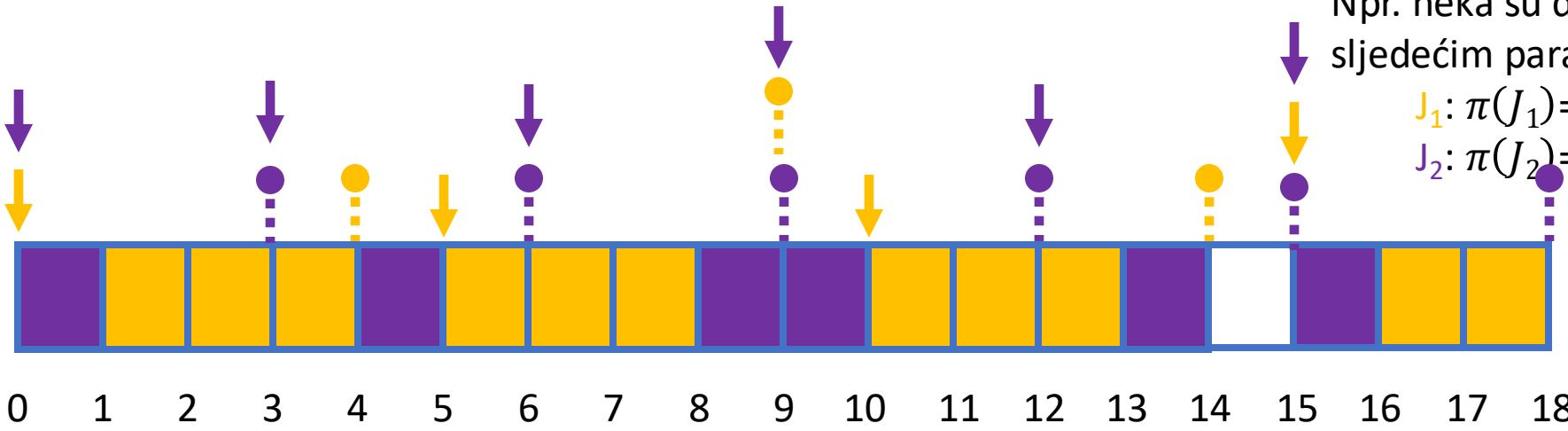
Politika najbližeg roka

- *Eariest deadline first* (EDF) politika:
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(t) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je rok $\delta(J_e)$ najbliži t: $\sigma(t) = J_e$
    - *Ako ima više poslova s istim najbližim rokom, odabir se vrši arbitrarno ili po alternativnom kriteriju*

Npr. neka su dani poslovi J_1 i J_2 sa sljedećim parametrima:

J_1: $\pi(J_1)$=5, $\delta(J_1)$=4, $\eta(J_1)$=3

J_2: $\pi(J_2)$=3, $\delta(J_2)$=3, $\eta(J_2)$=1



✓ rasporediv

...

J1 rok: za 3

J2 rok: za 3

Alternativni kriterij odabira (npr. manji id posla) !

Zadatak za vježbu: nacrtajte raspored nastao kao rezultat politike kod koje je alternativni kriterij daje prednost poslovima s većim id posla !

Svojstva politike najbližeg roka

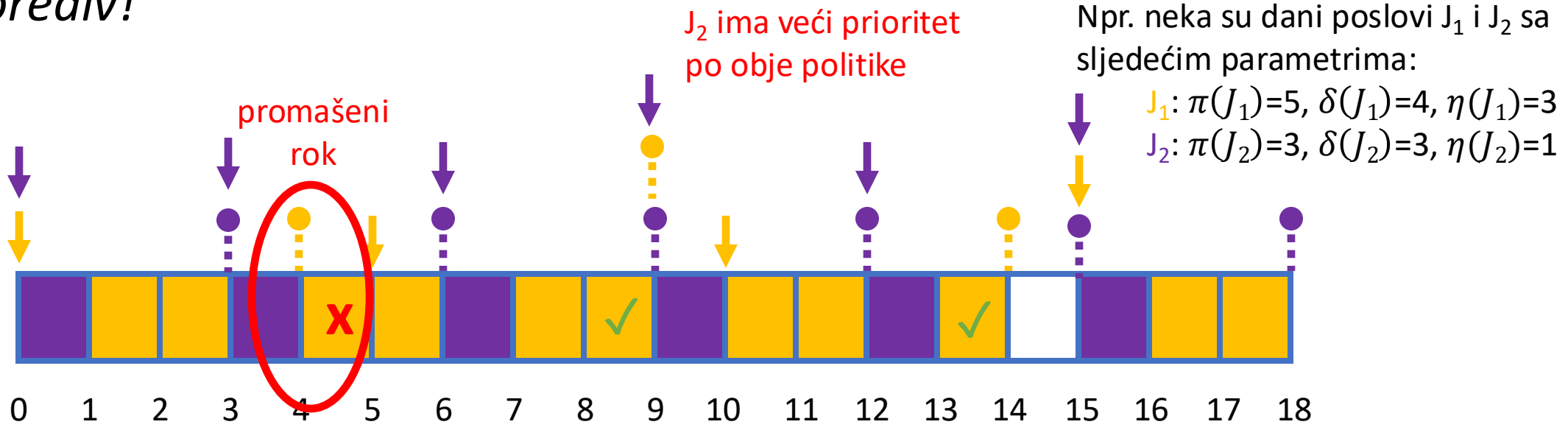
- Pripada u grupu tzv. *greedy* algoritama (odluke na osnovu lokalnih informacija, ne obraća se pažnja na globalne posljedice)
- Politika *dinamičkog prioriteta*
  - *Prioritet raspoređivanja posla J se mijenja u vremenu*
- Stvoreni raspored može biti *preemptive*
- Dokle god je model periodičkih poslova $\Im$ *rasporediv*, EDF politika će stvoriti raspored σ koji je sukladan svim rokovima poslova *J* iz $\Im$
- Brzi test rasporedivosti:
  - Svi rokovi su implicitni: $\delta(J) = \pi(J)$ --> instanca posla *J* mora završiti do dolaska nove instance
  - Model poslova $\Im$ je rasporediv ako vrijedi: $\mathrm{U}(\Im) \leq 1$

Politike fiksnog prioriteta poslova

- Poslovi ne mijenjaju prioritet tijekom vremena
  - Raspoređivač odlučuje o dodjeli vremenskog odsječka na osnovu prethodno definiranog, nepromjenljivog prioriteta posla
  - Politike određivanja prioriteta posla:
    - Prethodno, arbitrarno dodijeljen prioritet (npr. pozitivni cijeli broj)
    - *Deadline-monotonic* – poslovi s kraćim rokom imaju veći prioritet
    - *Rate-monotonic* – poslovi s kraćim periodom imaju veći prioritet
- Raspored posla σ se stvara za svaki vremenski odsječak *t* zasebno
  - na početku svakog vremenskog odsječka t $(t \geq 0)$ određuje se koji aktivni poslovi $J \in \Im$ još nisu završeni (još nisu odradili sve $\eta(J)$, trebaju procesorskog vremena)
  - ako nema takvih poslova: $\sigma(\mathrm{t}) = \perp$
  - ako ima takvih poslova, određuje se posao J_e čiji je prioritet najviši
    - *Ako ima više poslova s istim prioritetom, odabir se vrši arbitrarno, round-robin ili po nekom drugom alternativnom kriteriju*

Svojstva politika

- Poslovi više frekvencije dolaska imaju veći prioritet
  - Ako su krajnji rokovi *implicitni*, *rate-monotonic* politika je optimalna
    - U ovom slučaju su, u stvari, *deadline-* i *rate-monotonic* politike iste!
  - Ako su krajnji rokovi kraći od perioda dolaska poslova, *deadline-monotonic* je bolja
- Niti jedna od navedenih politika <u>ne garantira</u> pronalaženje rasporeda σ sukladnog svim rokovima poslova *J* iz $\mathfrak{J}$, makar model periodičkih poslova $\mathfrak{J}$ bio *rasporediv!*



Npr. neka su dani poslovi J_1 i J_2 sa sljedećim parametrima:

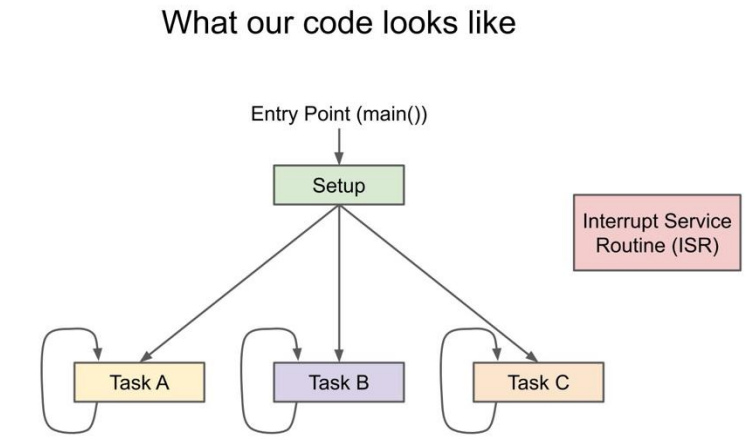
J_1: $\pi(J_1)=5$, $\delta(J_1)=4$, $\eta(J_1)=3$

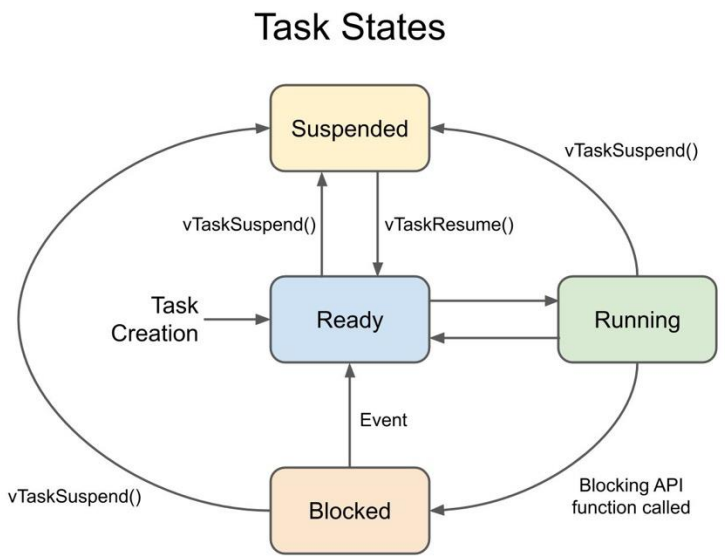
J_2: $\pi(J_2)=3$, $\delta(J_2)=3$, $\eta(J_2)=1$

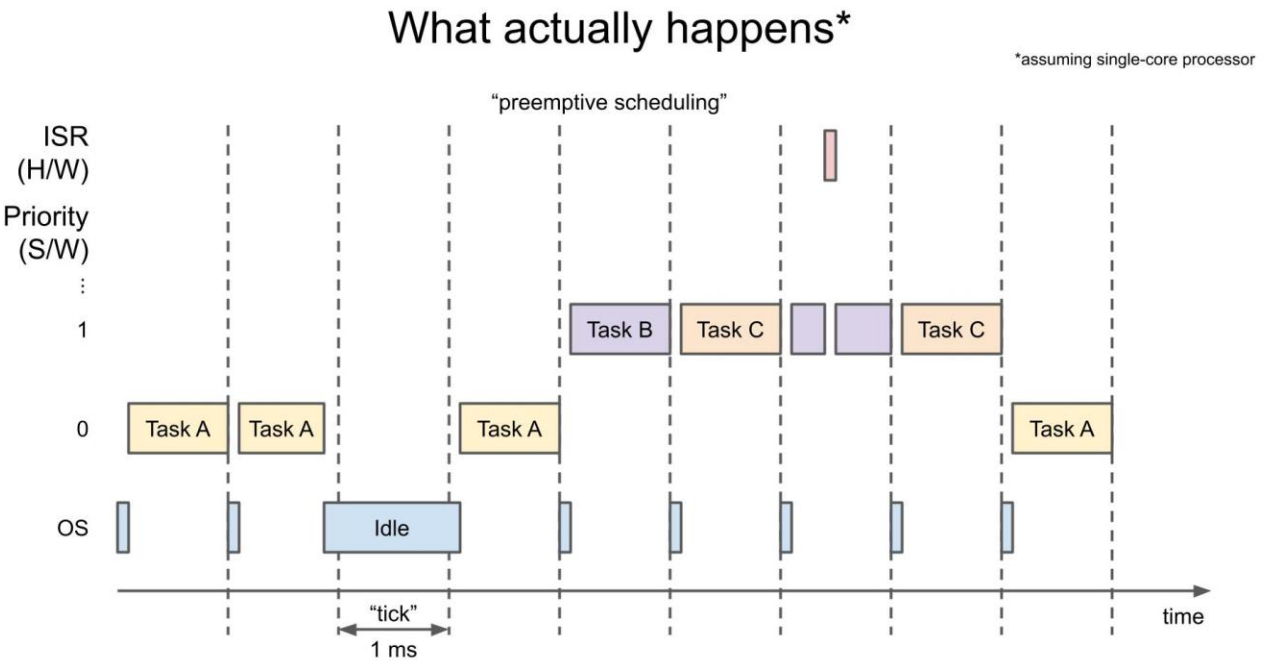
Svojstva politika

- Pripadaju u grupu tzv. *greedy* algoritama
- Stvoreni raspored može biti *preemptive*
- Minimalna potrebna procesna snaga prilikom određivanja rasporeda
  - Prioriteti se određuju *off-line*
  - U toku izvršavanja potrebno je samo poredati spremne poslove po prioritetu, odabrati onog s najvišim prioritetom

FreeRTOS na ESP32







https://www.digikey.ee/en/maker/projects/introduction-to-rtos-solution-to-part-3-task-scheduling/8fbb9e0b0eed4279a2dd698f02ce125f