

(2.3)

Mot: Pilc(t)

funtzio: Selvuentzatu: $\text{Pile}(t) \rightarrow \text{Selvuentzatu}(t)$

Ekuazioa:

$$\text{Selvuentzatu}(\text{PileHts}) = c \Rightarrow$$

$$\text{Selvuentzatu}(\text{PileHv}(x, q)) = \text{Selvuentzatu}(q) @ \langle x \rangle$$

(2.4)

Mot: Pilc(t)

funtzio: Despilletzeria: $\text{Pile}(t), xe \rightarrow \text{Pile}(t)$

$$\text{Despilletzeria}(\text{PileHts}, e) = \text{PileHts}$$

$$\text{Despilletzeria}(\text{PileHv}(x, q), e) = \begin{cases} \text{PileHv}(x, q) & \text{if } e = x \\ \text{Despilletzeria}(q, e) & \text{else} \end{cases}$$

(2.5)

Mot: Pilc(t)

funtzio: OrdosOrdolko: $\text{Pile}(t) \rightarrow \text{Pile}(t)$

Ekuazioa:

$$\text{OrdosOrdolko}(\text{PileHts}) = \text{PileHts}$$

$$\text{OrdosOrdolko}(\text{PileHv}(x, q)) = \begin{cases} \text{false} & \text{if } \cancel{\text{PileHv}(q)} \\ \text{true} & \text{if } x = \text{lehen}(q) \\ \text{OrdosOrdolko}(q) & \text{else} \end{cases}$$

(21)

Meth: Pile(\top)funkz: $Pile(\top) \times Pile(\top) \rightarrow Pile(\top)$

Erläuterung:

$$\text{Irculi}(PileHutse, PileHutse) = PileHutse$$

$$\text{Irculi}(PileHutse(P_{\cancel{x}, \cancel{q}}), PileHutse(x, q)) = PileHutse(x, q)$$

if PileHutse

Irculi(q, PileHutse(p, x)) else

$$\text{Irculi}(PileHutse, PileHutse(x, q)) = PileHutse(x, q)$$

$$\text{Irculi}(PileHutse(x, q), PileHutse(p, r)) = \begin{cases} & \\ & \end{cases} \text{Irculi}(q, PileHutse(x, p))$$

(22)

Meth: Pile(\top)funkz: Geinjektive: $Pile(\top) \times Pile(\top) \rightarrow Pile(\top)$

Erläuterung:

$$\text{Geinjektive}(PileHutse, PileHutse) = PileHutse$$

$$\text{Geinjektive}(PileHutse, PileHutse(x, p)) = PileHutse(x, p)$$

$$\text{Geinjektive}(PileHutse(x, q), PileHutse(x, q)) = PileHutse(x, q)$$

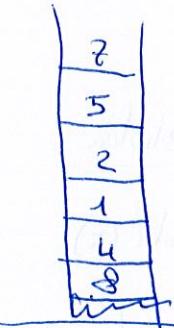
$$\text{Geinjektive}(PileHutse(x, q), PileHutse(s, p)) = \begin{cases} & \\ & \end{cases} \text{Irculi}(x, \text{Geinjektive}(q, PileHutse(s)))$$

~~Stabs~~

Meth.: Pile(τ)

~~Einführung:~~

Funktion: Pile(τ) \rightarrow Schwerpunkt(τ)



~~Einführung:~~

Schwerpunkt(PileHilfs) = $\langle \rangle$

~~Schwerpunkt(PileHilfs(P, x)) =~~

$$\begin{cases} \text{lass } \langle \rangle \text{ zu } P \text{ hinzufügen} \\ \text{oder } \text{Schwerpunkt}(P, x) @ \text{if } x \bmod 2 = 0 \end{cases}$$

~~Schwerpunkt(PileHilfs) = $\langle \rangle @ \langle \rangle$~~

~~Schwerpunkt(PileHilfs(P, x)) =~~

$$\begin{cases} \text{ordne } x @ \langle \rangle \text{ if } x \bmod 2 = 0 \\ \langle \rangle @ \text{ordne } x @ \text{else} \end{cases}$$

Meth.: Schwerpunkt(τ)

Funktion: ~~Nat~~ \times Schwerpunkt(τ) \rightarrow Schwerpunkt

~~Einführung:~~

Ordne($x, \langle \rangle$) = $\langle x \rangle$

Ordne($x, e.s$) = $\begin{cases} x @ e.s & \text{if } x \leq e \\ e.s @ \text{Ordne}(x, s) & \text{else} \end{cases}$

①

Selvantezie Erkunre

Motz: Arbolc(t)

Funktioe: Arbolc(t) \rightarrow Selvantezie(t)

Ekuozwelt:

Selvantezie Erkunre (hutse) = \leftrightarrow

$$\text{Selvantezie}_{\text{Erkunre}}(A, E_{\text{2U}}, E_{\text{3U}}) = \begin{cases} A @ \text{Selvantezie Erkunre}(E_{\text{2U}}) @ \text{Selvantezie}(E_{\text{3U}}) \\ \text{Baldin } \text{Hutse}(E_{\text{2U}}) \text{ and } \text{Hutse}(E_{\text{3U}}) \\ \text{Selvantezie Erkunre}(E_{\text{2U}}) @ \text{Selvantezie}(E_{\text{3U}}) \\ \text{Baldin } \text{Hutse}(E_{\text{2U}}) \end{cases}$$

②
a)

$$A = \text{Erratu}(1, \text{Erratu}(2, \text{Erratu}(3, \text{Erratu}(4)), \text{Erratu}(5, \text{Erratu}(6, \text{Erratu}(7, \text{Erratu}(8)))) \text{ hutse})$$

b)

$$f(A, 0) = f(\text{fill}(2, 5), 0) = f(2, 1) \text{ and } 0 \cdot f(5, 0) = f(\text{Erratu}(2, 3, 4), 1) - f(\text{Erratu}(5, \text{hutse}), 0)$$

$$= f(2, 1) \cdot f(3, 1) \cdot f(\text{hutse}, 1) \cdot f(6, 1) = f(\text{Erratu}(3, \text{hutse}, \text{hutse}), 1) \cdot \text{false} \cdot f(\text{Erratu}(7, 8), 1) =$$

$$= f(\text{hutse}, 1) \cdot f(\text{hutse}, 1) \cdot \text{false} \cdot f(2, 1) \cdot f(8, 1) = \text{false}, \text{false} \cdot \text{false}, f(\text{Erratu}(2, \text{hutse}, \text{hutse}), 1) \cdot$$

$$f(\text{Erratu}(2, \text{hutse}, \text{hutse})) = \text{false} \cdot \text{false} \cdot \text{false} \cdot \text{false} \cdot \text{false}.$$

2.6

pila bat, x elements bat die u Zaubohne, x agere u positionen

Motz: Pilc(t)

Lsg: Boolean

funtzio: agesprenpos: Pilc(t) \times Nat \times Positive \rightarrow Boolean

Erläuterl:

agesprenpos(~~Pilc(t), x, n~~, x, n) = False
Pile Hute

agesprenpos(Pilc(t), x, n) = $\begin{cases} \text{true} & \text{if } n=0 \text{ and } e=x \\ \text{agsprenpos}(q, x, n-1) & \text{else} \end{cases}$

2.7

Motz: Pilc(t)

funtzioa: Pilc(t), xe \times Pilc(t) \rightarrow Pilc(t) ~~\times Pilc(t)~~

Erläuterl:

Erekillo Erreki (Pilc(t), e, p) = $\begin{cases} \text{Pilc(t)} & \text{if } e=x \\ \text{Erekillo Erreki}(q, e, \text{Pilc}(x, p)) & \text{else} \end{cases}$

HurBeinoHandi: Selbstverständlichkeit

Inserenter / Erstellervorstellung erlaubt

$$\text{HurBeinoHandi}(\text{HerrBeinoHandi}(s)) = v + \text{HurBeinoHandi}(u)$$

Habsiechete: Behalde Habsiechete

HurBeinoHandi:

$$\begin{cases} u := s \\ v := 0 \end{cases}$$

$$\text{HurBeinoHandi}(1) \leftarrow \text{HurBeinoHandi}(0) = v$$

Ermittze:

Bilbesu:

$$\text{Hutsende}(u) \rightarrow \text{HurBeinoHandi}(u) = v + \text{HurBeinoHandi}(s) =$$
$$= v + \text{HurBeinoHandi}(0) =$$
$$= v + 0 = \boxed{v}$$

$$\text{Hutsende}(u) \rightarrow \text{HurBeinoHandi}(s) = v + \text{HurBeinoHandi}(u) =$$
$$= v + 0 = \boxed{v}$$

toleakete / der töles kote

$$\text{HurBeinoHandi}(s) = v + \text{HurBeinoHandi}(u) \approx$$

$$= v(1 + \text{HurBeinoHandi}(u)) = (\text{HurBeinoHandi}(\frac{\text{hutsende}(u)}{u}) + 1 + v =$$

$$= \text{HurBeinoHandi}(u) + v$$

hoso ueberse

7

while whilenen selber,
hoso while $\text{hutsende}(u)$ loop
end loop
return v

③

$$f(S, \langle \rangle) = S;$$

$$f(L, y, R) = y, R$$

$$f(x, S, y, R) = \begin{cases} x \cdot f(S, y, R) & \text{if } x \leq y \\ f(y, S, x, R) & \text{otherwise} \end{cases}$$

Inkrementale / Effektivitäts Erstellungszie Erklärung

$$f(\langle 1, 2, 3 \rangle, \langle 2, 3, 4 \rangle) = 1 \cdot f(\langle 2, 3 \rangle, \langle 2, 3, 4 \rangle) =$$

$$= 1 \cdot 2 \cdot f(\langle 3 \rangle, \langle 2, 3, 4 \rangle) = 1 \cdot 2 \cdot f(\langle 3 \rangle, \langle 2, 3, 4 \rangle) = 1 \cdot 2 \cdot f(2, \langle 3, 3, 4 \rangle) =$$

$$= 1 \cdot 2 \cdot 2 \cdot f(\langle \rangle, \langle 3, 3, 4 \rangle) = \cancel{f(2, 1, 2, 3, 3, 4)}$$

Effektivitäts Erstellungszie Erklärung.

$$f(S_1, S_2) = w @ f(u, v)$$

Hierarchie

$$w := \langle \rangle$$

$$S_1 := S_2$$

$$v := S_1$$

Basisfunktionen

↳ Befehl $u = \langle \rangle$

return $w @ v$

↳ Befehl $v = \langle \rangle$

return $w @ u$

TolesVek / DestolesVek

Bild: $\text{lehene}(v) \subseteq \text{lehene}(u)$

$$f(s_1, s_2) = w @ f(u, v) = w @ f(\text{lehene}(u), \text{lehene}(v)) =$$

$$= w @ (\text{lehene}(u) \cdot f(\text{lehene}(u), \text{lehene}(v))) =$$

$$= w' @ f(u', v')$$

Bild: $\text{lehene}(u) \supseteq \text{lehene}(v)$

$$f(s_1, s_2) = w @ f(u, v) = w @ f(\text{lehene}(u) \cdot \text{lehene}(v), \text{lehene}(u), \text{lehene}(v)) =$$

$$= w @ f(\text{lehene}(u) \cdot \text{lehene}(v), \text{lehene}(u), \text{lehene}(v)) =$$

$$= w' @ f(u', v')$$

Iterativ Dokumentation

function($s_1, s_2 : \text{Zeilige Schreibweise}(T)$) $\xrightarrow{\text{return: } e \in \text{Schreibweise}(T)}$

begin

$w := \langle ? \rangle;$

$u := \langle s_1 \rangle;$

$v := \langle s_2 \rangle;$

while $\text{?herrsche}(u)$ and $\text{?herrsche}(v)$ loop

if $\text{lehene}(u) \subseteq \text{lehene}(v)$ then

$w := w @ \text{lehene}(u);$

$@ u := \text{herrsche}(u);$

else

case $\text{lehene}(u)$ then $\text{log} := \text{lehene}(u);$

 Lafg: if $\text{lehene}(u)$ $u := \text{lehene}(v) @ u,$
 $\text{lehene}(u) := \text{lo}$ $v := \text{lehene}(u) @ v,$

Tolesket / Rest Tolesket

$$\begin{aligned}
 t(x.s) &= u @ f(\underset{R}{\cancel{\text{lehen}(R)}}) @ v = u @ f(\text{lehen}(R) \cdot \text{hondar}(R)) @ v = \\
 &= u @ u @ (\text{lehen}(R) \cdot f(\text{hondar}(R)) @ \text{lehen}(R) @ v = \\
 &= u @ f(R) @ v
 \end{aligned}$$

with Toleranz

funktion $f(S : \text{Schwartz}(T)) \rightarrow \text{return } e : \text{Schwartz}(T)$

begin

$R := S;$

$U := R;$

$W := \perp;$ $\text{Hilfe_Dc}(R) \leftarrow$

while $\text{hilfe_Dc}(R) \neq \perp$ loop

if $\text{hilfe_Dc}(R) > 1$ then

~~$e := u @$~~

$u := u @ \text{lehen}(R);$

$r := \perp @ \text{Toleranz}(R)$
 $\text{lehen}(R) @ r;$

$e := v @ f(\text{hondar}(R)) @ r;$

else

$e := u @ (\text{lehen}(R) \neq @ r);$

end if

end loop;

return $e;$

③

Burststellen Methoden:

$$f: \text{Schwierigkeit}(\tau) \rightarrow \text{Schwierigkeit}(t)$$

$$f(L) = \text{erstree}$$

$$f(x) = x$$

$$f(x \cdot s) = x \cdot f(s) @ L_x$$

Invertionsregel / Kettentheorie & Rechenregeln

$$f(1, 2, 3, 4, ?) = 1 \cdot f(2, 3, 4, ?) @ L_1 = 1 @ f(2, 3, 4, ?) @ L_1 =$$

$$= L_1 @ (2 \cdot f(3, 4, ?)) @ L_2 @ L_1 = 1, 2 @ (f(3, 4, ?)) @ L_2, 1 =$$

$$= 1, 2 @ (3 \cdot f(4)) @ 3 @ L_2, 1 = 1, 2, 3 @ f(4) @ L_3, 2, 1 =$$

$$= 1, 2, 3, 4, 3, 2, 1$$

$$f(s) = u @ f(r) @ v$$

Häufigkeiten

$$R := S$$

$$u := L$$

$$v := C @ L$$

Bilanzierung

$$\text{Baldin Luzece}(R) \geq 1$$

$$c := \text{d. lehene}(R) @ f(\text{Handelse}(R) @ \text{lehene}(R))$$

$$\text{d. Bestell. Luzece}(R) = 1$$

$$e := \text{lehene}(R)$$

80

Mote; Schubert(?)

funkzioe: Selbstverständl x Nat \rightarrow Selbstverständl (+)

Ekuaziole:

$$\text{Aunizhi}(\langle \rangle, n) = \langle \rangle$$

Aurizkhi(e, S, n) = $\begin{cases} \langle \text{ } \rangle & \text{if } n=0 \\ e \text{ } \text{Aurizkhi}(S, n-1) & \text{else} \end{cases}$

4.11

Mota: Schmetterlinge (5)

funtzioc:

PileBilwutq: Schwerpunkt (τ) \rightarrow Pile (τ)

Elwoodwale.

pile Bihurku($c >$) = pilektutse, pihurku($c < s$)

$$\text{pilcBikurtu}(e, s) = \begin{cases} \text{pilcBikurtu}(p, e) & \text{if } e \bmod 2 = 0 \\ \text{pilaBikurtu}(s) & \text{else} \end{cases}$$


```
public Boolean Zentraleide (Individ bestetruide) {  
    if (this.xWoord == bestetruide.xWoord) && this.yWoord == bestetruide.yWoord) &&  
        this.z == bestetruide.zWoord) {  
    return true; }  
    else {  
        return false;  
    }  
  
public Boolean InstruktiveEsfera (Entfer E) {  
    return (this.alde  $\leq$  E * 2 * (E - Entfer)) && this.Zentraleide(E);  
}
```

```
public abstract class Irudi3D {
```

```
    private double z;           throws IOException  
    public Irudi3D (double z) { ... }  
    public double zHond () { ... }  
    double
```

```
    public final double distantzia () { ... }
```

```
    public final boolean kontrolidea (Irudi3D batekIrudi) { ... }
```

```
    public abstract boolean institribueEstera (Estera e) { ... }
```

```
    public abstract Bolumene ();
```

```
{
```

```
public class Estera extends Irudi3D {
```

```
    private double erradua;
```

```
    public Estera () throws & IOException { ... }
```

```
    public double azalea () { ... }
```

```
b) public Boolean institribueEstera (Estera e) { ... }  
    Boolean
```

```
    public double Pbolumene () { ... }
```

```
    public double Erradua () { ... }
```

```
{
```

```
b)
```

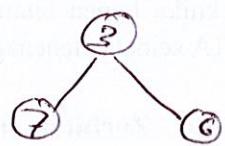
```
    public Boolean institribueEstera (Estera e) {
```

```
        return (this.erradua <= e.erradua) && this.kontrolidea (e);
```

```
{
```

$\text{Errortu}(3, \text{Hutse}, \text{Hutse}) \rightarrow \textcircled{3}$

$\text{Errortu}(3, \text{Errortu}(2, \text{Hutse}, \text{Hutse}), \text{Errortu}(2, \text{Hutse}, \text{Hutse})) \rightarrow$



Datu-Hut. | Funktion erreichbarkeit

Selbstz. (T)

< >

e, s

Pile (T)

pilchutse

pilchutu (P, x)

Arbitr. (T)

Hutse

Errortu(e, e&h, e&l)

e&h, e&l & Arbitr(T) e, x & T

s & Selbstz. (T)

P & Pile (T)

Eckziegel

Hutse - D_c (Errortu(e, e&h, e&l)) = False

Hutse - D_c (Hutse) = True

Nodoll(Hutse) = 0

Nodoll(Errortu(e, e&h, e&l)) =

~~e&h, e&l~~

Nodoll(e&h), Nodoll(e&l) = 1 + Nodoll(e&h) + Nodoll(e&l)

Teknika honen bitartez, prozesadoretik hurbilen dauden periferikoak dira lehentasun gorenekoak (INTA seinalea lehenago hartzen baitute). Ikusi 3.13 irudia.

3.2.2.4 Zerbitzu-errutinaren exekuzioa.

Exekutatu behar den zerbitzu-errutina zein den aztertu ondoren, burutu behar diren eragiketak periferikoaren eta lortu nahi den emaitzaren menpekoak dira.

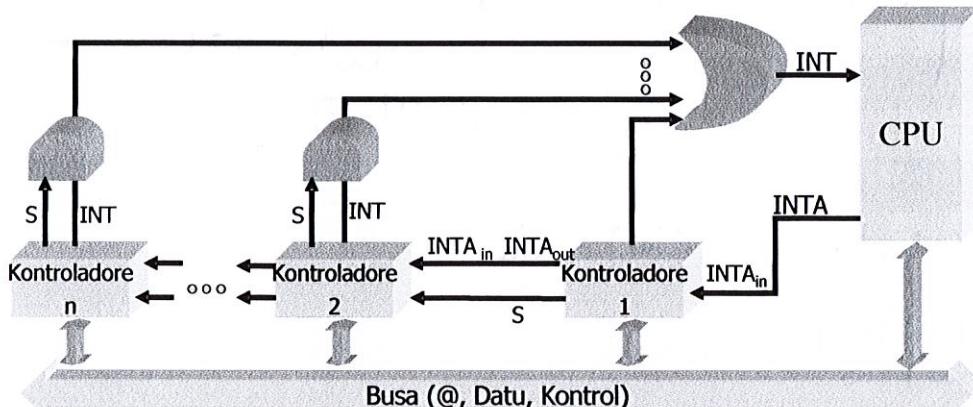
3.2.2.5 Etendako programaren egoera berreskuratu.

Zerbitzu-errutinaren azken agindua itzuliarena da. Itzulera aginduak etendako programaren egoera (PC eta PSW) berreskuratzen du bere exekuzioa berriro has dadin.

3.2.2.6 Maila anitzeko etenak.

Konputagailu gehienetan CPU eten daiteke aurreko eten baten tratamendurako zerbitzu-errutina exekutatzeko ari den bitartean. Kasu honetan, zerbitzu-errutina berrira jauzten da eten berria aurrekoa baino lehentasun handiagokoa bada. Beraz, n lehentasuna duen etena sortzen denean, kasu egingo zaio etenak baimenduta badaude eta momentu horretan lehentasun handiagoko zerbitzu-errutinaren bat exekutatzeko ari ez bada. Hau da, lehentasun mailak ezartzen dira. Kasu honetan IF ez da 0-ra jarriko automatikoki.

3.14 irudian, lehen ikusitako **Daisy-Chain** hardwarearen eskeman, maila anitzeko etenekin lan egin ahal izateko egin beharreko aldaketak zeintzuk diren ikus daiteke.



3.14 Irudia: Daisy-chain hardwarea maila-anitzeko etenak jasateko aldaketekin

K_i kontroladoreak, etena eskatu duenak hain zuzen, INTA seinalea hartzen duenean CPU-ri zerbitzu-errutina identifikatzeko behar duen informazioa bidaltzen dio eta S_i seinalea desaktibatzen du; honela adierazten du berari dagokion zerbitzu-errutina exekutatzeko ari dela. Orain K_{i+1} kontroladoreak etena eskatzen badu INTA seinalea ez zaio iritsiko S_i seinalea desaktibatuta dagoenez prozesadorera iristen den INT seinalea ez baita aktibatuko; beraz, lehentasuna errespetatzen da. Etena K_{i-1} kontroladoreak eskatzen badu berriz zerbitzatuko da. Zerbitzu-errutinaren exekuzioa amaitzen denean S_i seinalea aktibatu beharko da.

3.2.2.7 Etenen kontroladorea.

Etenen kontroladorea CPU eta S/I-ko kontroladoreen arteko bitartekaria da, bera dela medio etenen kudeaketa eraginkorra lortzen delarik. Eten-eskaerak hartzen ditu eta lehentasunaren arabera erabaki

①

Note: Schluesselwort (I)

Funktion: Ordenduo: Schluesselwort (I) x Schluesselwort (T) \rightarrow Schluesselwort (T)

Elaboration:

Ordenduo (e_1, S , e_2, S , e_3, S) = $e_1, S \sqsubset e_2, S$

Ordenduo ($<$, $<$) = $<$

Ordenduo ($<$, $< e_2, S$) = $< e_2, S$

Ordenduo ($< e_1, S$, $< e_2, S$) = if $e_1 \neq e_2$ then

$e_1, S \sqsubset e_2, S$ @ Ordenduo (S, e_1, S , e_2, S)

else $e_1, S @ Ordenduo (S_1, e_2, S)$

$e_2, S \sqsubset e_1, S$ @ Ordenduo (e_1, S , e_2, S)

② Note: Schluesselwort (I) der Pile (t) $S_1, S_2, S \in$ Schluesselwort

Note: Tag: Boolean

Funktion: Ordenein: Schluesselwort (T) x pile (t) \rightarrow Boolean

Elaboration: Ordenein ($<$, \boxed{S}) = true

Ordenein ($\boxed{S}, <$) = false

Ordenein ($<$, \boxed{P}) = false

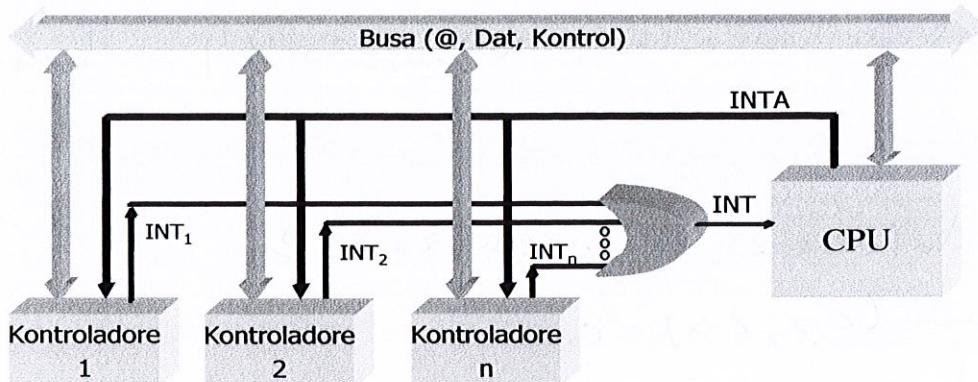
Ordenein ($< e_1, S \rightarrow \boxed{P}$) =

Ordenein ($< e_1, S$, pile $\boxed{h(t)}$) = true

Ordenein ($< e_1, S$, pile $\boxed{h(t)}$) = false

Ordenein ($< e_1, S$, pile $\text{def}(P, x)$) = false

Ordenein ($< e_1, S$, pile $\text{def}(P, x)$) = $\begin{cases} \text{Ordenein}(\$, P) & e_1 \neq x \\ \text{best false} & \text{bestel.} \end{cases}$



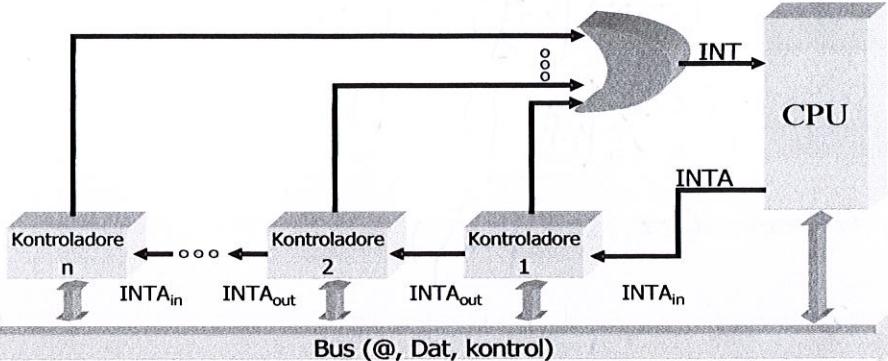
3.12 Irudia: kontroladorearen identifikazioa egiten duen hardwarearen eskema

CPU-k eten-eskaera onartzen duenean aktibatu den sarrerari dagokion INTA irteera aktibatzen du. Seinale hau kontroladoreen sarrera izango da. Etena eskatu duen periferikoak seinale hau aktibatu dela detektatzean, CPU-ri informazioa bidaliko dio honek jakin dezana zein den exekutatu behar duen zerbitzu-errutina.

Kontroladoreak prozesadoreari bidaltzen dion informazioa ondokoetako bat izan daiteke:

- Zerbitzu-errutinaren helbidea (SIGNETICS 2650).
- Prozesadoreak exekutatu behar duen aginduaren eragiketa-kodea; agindu hau, logikoki, zerbitzu-errutinarako jauzia da (i8080).
- Prozesadoreak exekutatu behar den zerbitzu-errutinaren helbidea gordetzen duen taula bat atzitzeko erabiliko duen identifikadorea. Teknika honi "**eten bektorizatua**" izena ematen zaio (i8086).

Edozein kasutan ere, bigarren arazoa ebazteke daukagu, hots: zer egin une berean periferiko batek baino gehiagok eten-eskaera sortzen badu? Aurreko eskemari ez badiogu ezer gehitzen, etena eskatu zuten periferiko guztiekin, INTA seinalea detektatzean, dagokien informazioa bidali nahi izango liokete prozesadoreari, guztiekin batera! Arazo hau ebazteko erabiltzen den aukeretako bat ikusiko dugu, **Daisy-chain edo margarita-katea**.



CPUari kontrolagailu bat baino gehiagoren eten eskaera heltzen bazaio, INTA sarrera bat baino gehiago egongo dira aktibatuta. Horrela CPUak banan banan aztertuko ditu kontrolagailu guztiak, aztertutako kontrolagailuak eten-eskaerarik luzatu ez badu, orduan kontrolagailu horren INTA irteera aktibatuko da, eta hurrengo kontrolagailua azterzera pasatuko da.

Eten-eskaera luzatu duen kontrolagailura heltzen denean, CPUari exekuzio errutinaren informazioa bidaliko dio.

Bi selventeia ordinato emende schentare ordinati balle
italtor Iva fntdior

$\langle X, X, 2, 4 \rangle \quad \langle X, X, 5, ? \rangle$

function Ordento-Selventoile (S₁: in String, S₂: in String) S : string return String

begin

if S₁'length = 0 then

return ~~S₂~~ S₂'(First.. S₂'Last);

elsif S₂'Length = 0 then

return ~~S₁~~ S₁'(S₁'First.. S₁'Last);

else

if S₁'First < S₂'First

return S₁'First & Ordento-Selventoile (S₁'First+1.. S₁'Last, ...);

else

return S₂'First & Ordento-Selventoile (S₁'(S₁'First.. S₁'Last), S₂'(First+1..

end if;

end if;

S = $\langle 1, 1, 2, 3, 4, 5, ? \rangle$

Suposatuko dugu, etenaren zerbitzu-errutinaren exekuzioa hasten denetik amaitu arte beste eten- eskaerak galarazita daudela. Hau da, CPU-k eten bat zerbitzatuko duela erabakitzten duenean, IF bita automatikoki Ora jarriko du eta ez da berriro aktibatuko zerbitzu-errutina amaitu arte.

3.2.2.2 Gorde etendako programaren egoera.

Teorikoki etenaren zerbitzu-errutinaren exekuzioa amaitzen denean etendako programaren exekuzioa berreskuratu behar da. Hau egin ahal izateko beharrezkoa da programaren egoera gordetzea etena onartzen denean.

Automatikoki gorde behar den gutxienezko informazioa hauxe da:

- PC: zerbitzu-errutinatik bueltatzerakoan exekutatu behar den hurrengo agindua zein den jakiteko.
- PSW (baldintza-bitak edo egoera-bitak): zerbitzu-errutinatik bueltatzerakoan, hau exekutatzen hasi aurretik zegoen egoera berean jarraitu dezan.

Zerbitzu-errutinak, bere aldetik, berak erabiltzen dituen erregistroen edukia gorde beharko du. Zerbitzu-errutina ezin dela eten esan dugunez (maila bakarreko etenak), prozesadorearen egoera posizio finkotan gorde daiteke. Geroxeago ikusiko dugu prozesadore gehienetan maila anitzeko etenak onartzen direla, eta horrexegatik prozesadorearen egoera pilan gorde beharko dela.

3.2.2.3 Zerbitzu-errutina edo periferikoaren identifikazioa.

Etenak sor ditzaketen periferikoak desberdinak direnez eta bakoitzak tratamendu egokitua behar duenez, etena detektatu (eta onartu) ondoren beharrezkoa da periferiko eten-sortzailea identifikatzea berari dagokion zerbitzu-errutinara jauzteko. Arazo hau agertzen da periferiko batzuk eten-sarrera bakarra erabiltzen dutenean. Arazo hau ebazteko ondoko galderak egin behar dizkiogu gure buruari:

- nork sortu du etena? Identifikazioa hardware bidez zein software bidez egin daiteke.
- etena periferiko batek baino gehiagok eskatzen badu une berean, zeini egingo zaio kasu lehenik? Lehentasunak ezarri behar dira.

Software bidezko identifikazioa:

Etenak baimenduta daudela, INT seinalea aktibatzen den bakoitzean, CPU zerbitzu-errutina konkretu batetara jauzten da (zerbitzu-errutina bana eten-sarrerarako): **zerbitzu-errutina orokorra**. Errutina honen lehenengo aginduen helburua hauxe da: eten-eskaera sortu duen periferikoaren identifikazioa egitea, inuesta baten bidez, kontroladoreen egoera-erregistroak aztertuz. Inkestaren ordenak periferikoen lehentasuna adierazten du. Periferikoa identifikatua izan denean, berari dagokion zerbitzu-errutinara jauzten da.

Ebazpen hau oso simplea da baina motelegia izan daiteke. Hau erabiltzen duen makina bat: Nintendo DS.

Hardware bidezko identifikazioa:

Kasu honetan periferikoa bera identifikatzen da prozesadorearen aurrean. Horretarako, CPU-k irteera bereziak baditu eten eskaerak onartzeko, INTA hain zuzen (etenaren onarpena), 3.12 irudian ikus daitekeen bezala. Irteera hauen kopurua INT sarreren kopuru bera da $[INT_1, \dots, K] ==> INTA_1, \dots, K]$.

1.3

Mote: Arbit(T)

Lag: not

Erf. Mete:

Hostokop: $\text{arbit}(T) \rightarrow \text{not}$

Erlverziale:

① Hostokop(utsc) = 0

② Hostokop(e, esl, esl) = $\begin{cases} 1 & \text{baldin } \cancel{\text{Hostokop}(esl)} - \cancel{\text{Hostokop}(esl)} \\ & \text{Hostokop}(esl) + \text{Hostokop}(esl) \text{ bæst}, \\ & \cancel{\text{Hostokop}(esl)} \end{cases}$

③

Esl etc esl addebagi kopenne ① baldin bæði eru addebagilag (esl, esl),
grítili bæti gefnivea bat idan bæða d.

④

Elemento har arbóla tilte bætan agerðan den addi kopenne
kalkvéltaðaðu fyrir fyrir.

Mote: Arbit(T)

Lag: Net

Erlverziale:

Agerkop: $\text{TxArbit}(T) \rightarrow \text{not}$

Erlverziale:

Agerkop(x, utsc) = 0

Agerkop($x, \text{Enda}(c, \text{Esl}, \text{esl})$) = $\cancel{x} \begin{cases} 1 & \text{Agerkop}(x, \text{esl}) + \text{Agerkop}(x, \text{esl}) \\ & \cancel{\text{Agerkop}(x, \text{esl})} + \text{Agerkop}(x, \text{esl}) \\ & \cancel{\text{Agerkop}(x, \text{esl})} + \text{Agerkop}(x, \text{esl}) \end{cases}$

behar du heldutako eskaera CPU-ri luzatu behar zaion ala ez. Eten-eskaera CPU-k onartzen duenean, etenen kontroladoreak zerbitzu-errutina identifikatzeko behar den informazioa bidaltzen dio CPU-ri.

Etenen kontroladorearen beste aukerak hauexek dira:

- etenak bereiztuta maskaratzea,
- lehentasunak programatza (dinamikoki),
- etenen kontroladoreak kateatzea.

Bi arbolak erandie bat bestearren is pilur

