طراحی سیستم‌های تعبیه‌شده
Embedded System Design

فصل دوم ـ قسمت دوم

مشخص‌سازی
Specifications

کاظم فولادی
دانشکده‌ی مهندسی برق و کامپیوتر
دانشگاه تهران

kazim@fouladi.ir

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی

۱ ـ زبان‌های همگام و ناهمگام

توصیف پردازش‌های متعدد در بسیاری از زبان‌ها غیرقطعی است:

ترتیب اجرای وظایف قابل اجرا تعیین نمی‌شود (می‌تواند بر نتیجه مؤثر باشد).

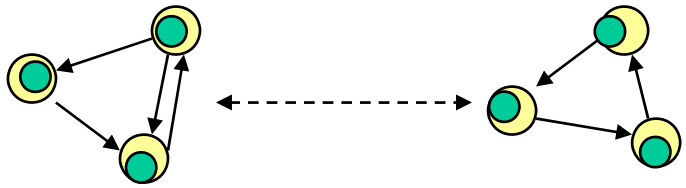
زبان‌های همگام: بر اساس مدل‌های آتاماتا

«هدف زبان‌های همگام در فراهم کردن ساختارهای پیمانه‌ای سطح بالا، ساده‌تر کردن طراحی چنین آتاماتونی است [Halbwachs].»

«زبان‌های همگام آتاماتون‌های همروند را توصیف می‌کنند . . .
وقتی آتاماتون‌ها به طور موازی ترکیب می‌شوند، یک گذر آن از گذرهای «همزمان» همه‌ی آنها تشکیل می‌شود. »

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی

۱ ـ زبان‌های همگام و ناهمگام (ادامه)



زبان‌های همگام به طور ضمنی فرض می‌کنند که یک سیگنال **کلاک** (سراسری) وجود دارد. در هر تیک کلاک، همه‌ی ورودی‌ها بررسی می‌شوند، خروجی‌ها و حالت‌های جدید محاسبه می‌شوند و سپس گذرها انجام می‌شوند.

این مستلزم مکانیزم پخش همگانی برای همه‌ی اجزای مدل است.

این یک نگاه ایده‌آل به همروندی است که مزیت آن **تضمین رفتار قطعی** است.

StateCharts یک زبان همگام است.

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۲ ـ ویژگی‌های پردازش‌ها

- **تعداد پردازش‌ها**

  ایستا

  پویا (معماری سخت‌افزاری متغیر به طور پویا؟)

- **تودرتو بودن**

  – اعلان تودرتوی پردازش‌ها

```
process{
    process {
        process {
}}}
```

  – اعلان پردازش‌ها در یک سطح

```
process { … }
process { … }
process { … }
```

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۲ ـ ویژگی‌های پردازش‌ها (ادامه)

- تکنیک‌های مختلف برای ایجاد پردازش‌ها
  - تشریح دقیق در کد منبع برنامه (مانند ADA)

```
declare
   process P1 ...
```

  - دستورات صریح fork و join (در یونیکس)

```
id = fork();
```

  - فراخوانی‌های سیستم برای ایجاد پردازش

```
id = create_process(P1);
```

StateCharts حاوی تعداد ایستایی از پردازش‌ها، اعلان تودرتوی پردازش‌ها است و روش ایجاد پردازش‌های آن از طریق تشریح دقیق در کد برنامه می‌باشد.

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۳ ـ پارادایم‌های ارتباطات

- **انتقال پیام**
  - **ارتباطات مسدودنشونده**

فرستنده لازم نیست صبر کند تا پیام به دست گیرنده برسد.

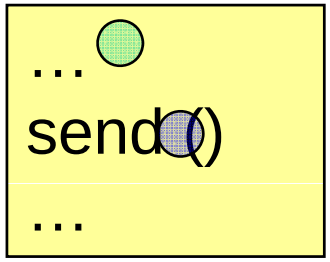
مساله‌ی بالقوه: سرریز بافر

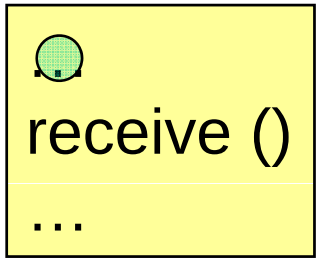
non-blocking communication

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۳ ـ پارادایم‌های ارتباطات (ادامه)

– ارتباطات مسدودشونده، ارتباطات مبتنی بر قرار ملاقات

فرستنده صبر می‌کند تا اینکه گیرنده پیام را دریافت کند.





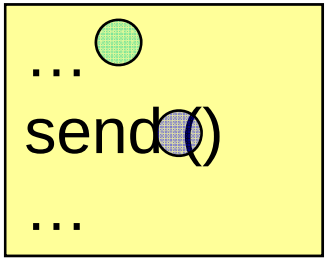
blocking communication, rendez-vous

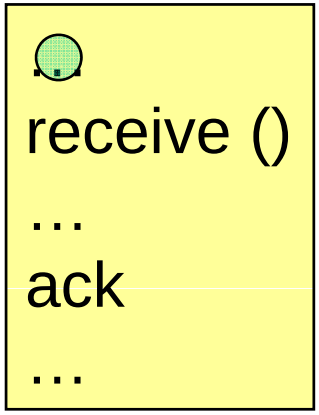
برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۳ ـ پارادایم‌های ارتباطات (ادامه)

- **قرار ملاقات توسعه‌یافته**

دریافت صریح سیگنال acknowledge از گیرنده لازم است.

گیرنده می‌تواند پیش از ارسال acknowledge پیام را بررسی کند.





برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی

۳ ـ پارادایم‌های ارتباطات (ادامه)

- **حافظه‌ی اشتراکی**

متغیرها برای چند پردازش دسترس‌پذیر هستند.

امکان بروز شرایط مسابقه (race condition) : نتایج ناسازگار

نواحی بحرانی = بخش‌هایی از کد که در آنها باید دسترسی انحصاری به منبع اشتراکی r (مانند حافظه‌ی اشتراکی) تضمین شود.

```
process a {
  ..
  P(S)  //obtain lock
  ..        // critical section
  V(S)  //release lock
}
```

```
process b {
  ..
  P(S)  //obtain lock
  ..        // critical section
  V(S)  //release lock
}
```

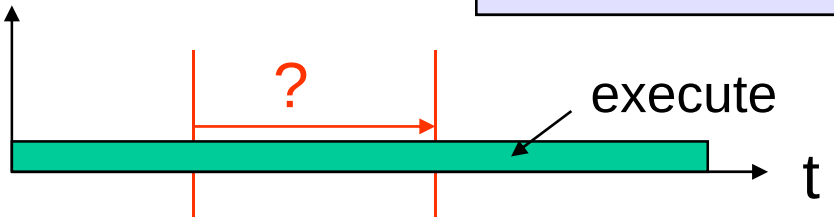
دسترسی عاری از مسابقه
به کمک **سمافور** S

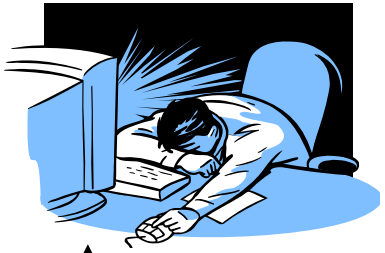
StateCharts برای ارتباطات میان پردازش‌ها از حافظه‌ی اشتراکی استفاده می‌کند

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
٤ ـ مشخص کردن زمان

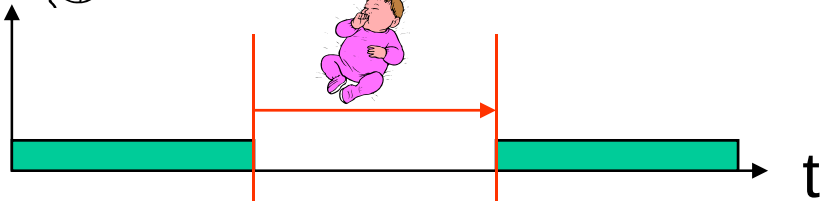
چهار نوع مشخص‌سازی زمانی لازم: [Burns, 1990]

- **اندازه‌گیری زمان سپری‌شده**
  بررسی اینکه از آخرین فراخوانی چه مدتی گذشته است.



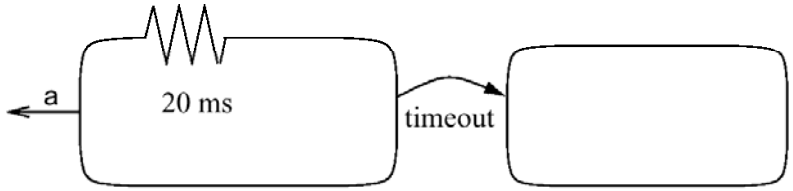


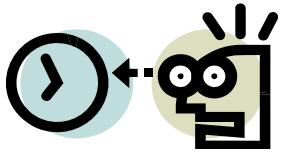
- **ابزارهایی برای پردازش‌های تاخیردار**



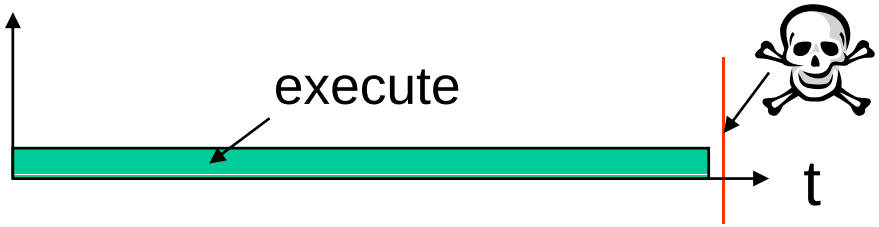
برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
٤ ـ مشخص کردن زمان (ادامه)

- **امکان مشخص کردن time-outها**
  ماندن در یک حالت به اندازه‌ی یک حداکثر زمانی





- **روش‌هایی برای مشخص کردن مهلت زمانی**
  وجود ندارد یا اینکه باید در فایل‌های کنترلی مشخص شود.



StateCharts حاوی مکانیزمی برای مشخص کردن time-out است.
سایر مشخصه‌های زمانی در آن پشتیبانی نمی‌شود.

برخی از ویژگی‌های عمومی زبان‌های مشخص‌سازی
۵ ـ استفاده از دستگاه‌های I/O غیر استاندارد

دسترسی **مستقیم** به سوییچ‌ها، نمایشگرها و . . .

حفاظت لازم نیست؛ سیستم عامل می‌تواند سریع‌تر عمل کند.

در StateChart پشتیبانی نمی‌شود.

SDL

Specification and Description Language

زبان طراحی شده برای مشخص‌سازی سیستم‌های توزیع‌شده (distributed)

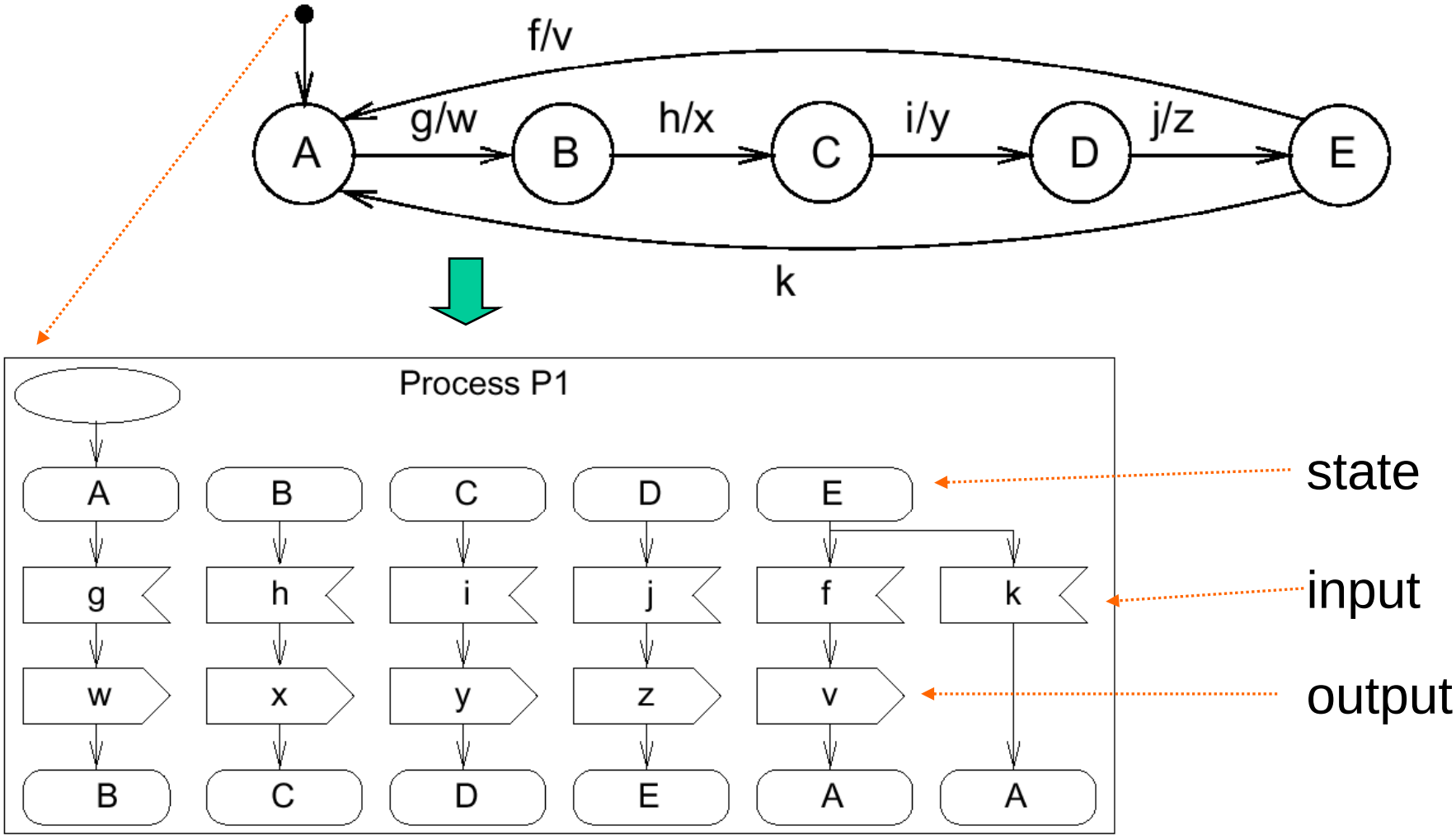
- به اوایل دهه‌ی ۱۹۷۰ برمی‌گردد.
- مفاهیم رسمی آن در اواخر دهه‌ی ۱۹۸۰ تعریف شد.

- Defined by ITU (International Telecommunication Union):
  Z.100 recommendation in 1980
  Updates in 1984, 1988, 1992, 1996 and 1999

SDL

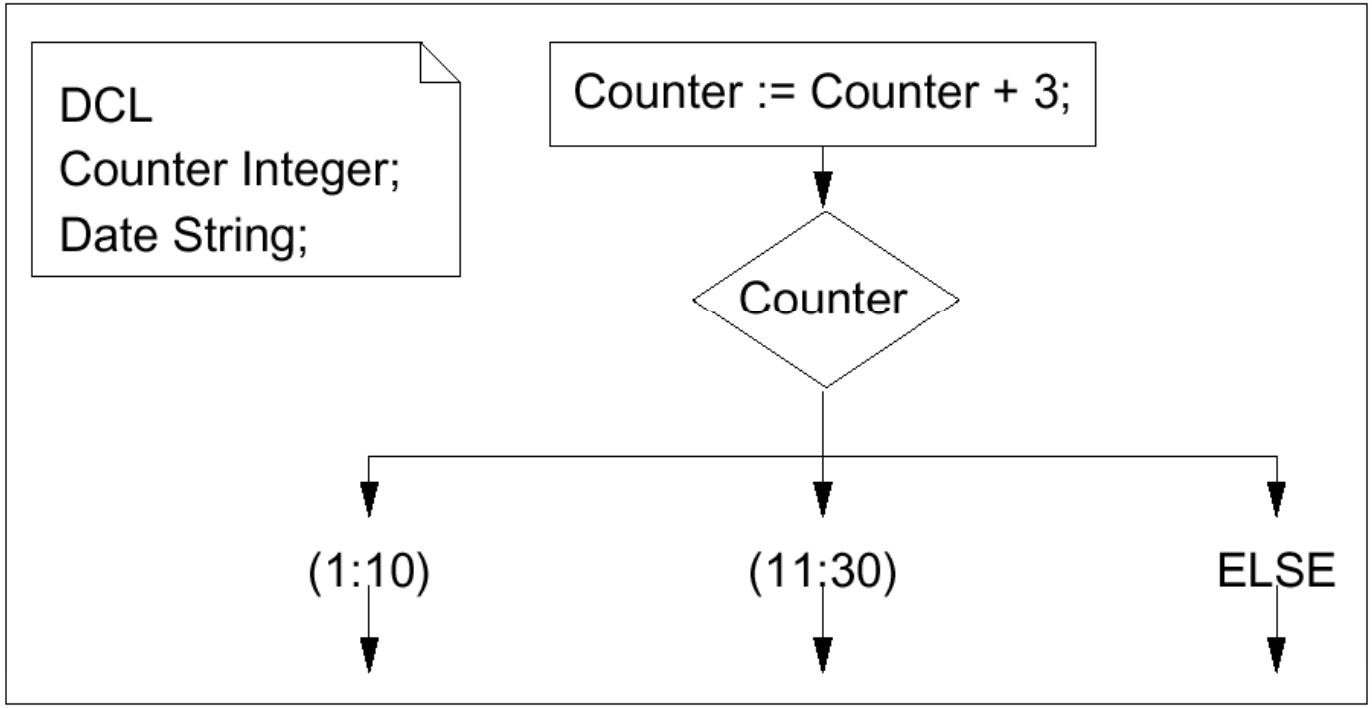
- فراهم کردن قالب‌های متنی و گرافیکی برای جلب نظر همه‌ی کاربران
- همانند StateCharts مبتنی بر مدل محاسباتی CFSM است:
  هر FSM یک **پردازش** نامیده می‌شود.
- اما، برای ارتباطات به جای حافظه‌ی اشتراکی
  از **انتقال پیام ناهمگام** استفاده می‌کند.
- SDL از عملیات بر روی داده‌ها پشتیبانی می‌کند.

بازنمایی SDL برای پردازش‌ها/FSMها



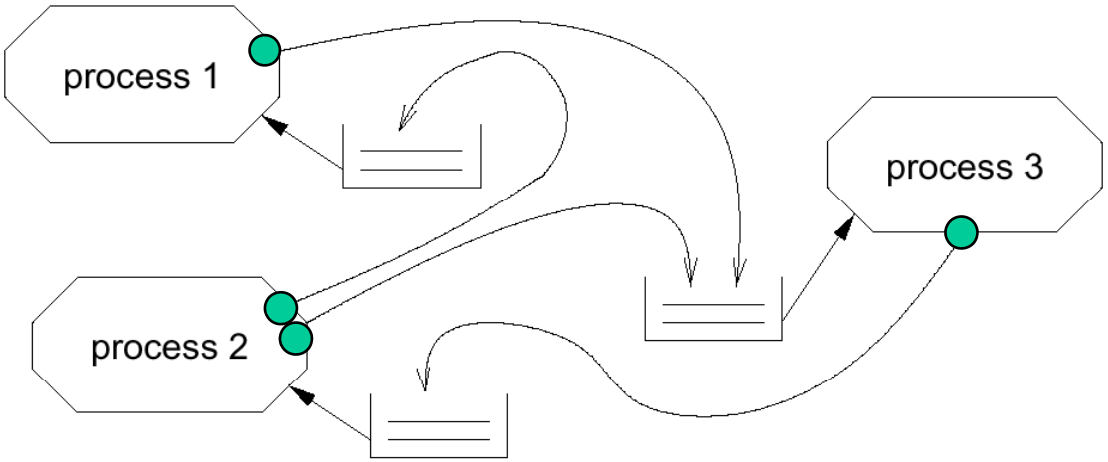
SDL: عملیات بر روی داده‌ها

متغیرها می‌توانند به طور محلی برای پردازش‌ها اعلان شوند.
نوع آنها می‌تواند از پیش‌تعریف شده باشد یا در SDL تعریف شود.
SDL انواع داده‌ی انتزاعی (ADT) را پشتیبانی می‌کند. مثال:



ارتباطات بین FSMهای SDL

ارتباطات میان FSMها (یا پردازش‌ها) مبتنی بر انتقال پیام است؛
با این فرض که صف FIFO به طور بالقوه بی‌نهایت بزرگ است.



- هر پردازش درایه‌ی بعدی را از FIFO واکشی می‌کند،
- بررسی می‌شود که آیا ورودی گذری را فعال می‌سازد یا خیر،
- اگر بله: انجام گذر،
- اگر خیر: نادیده گرفتن ورودی (استثنا: مکانیزم SAVE)

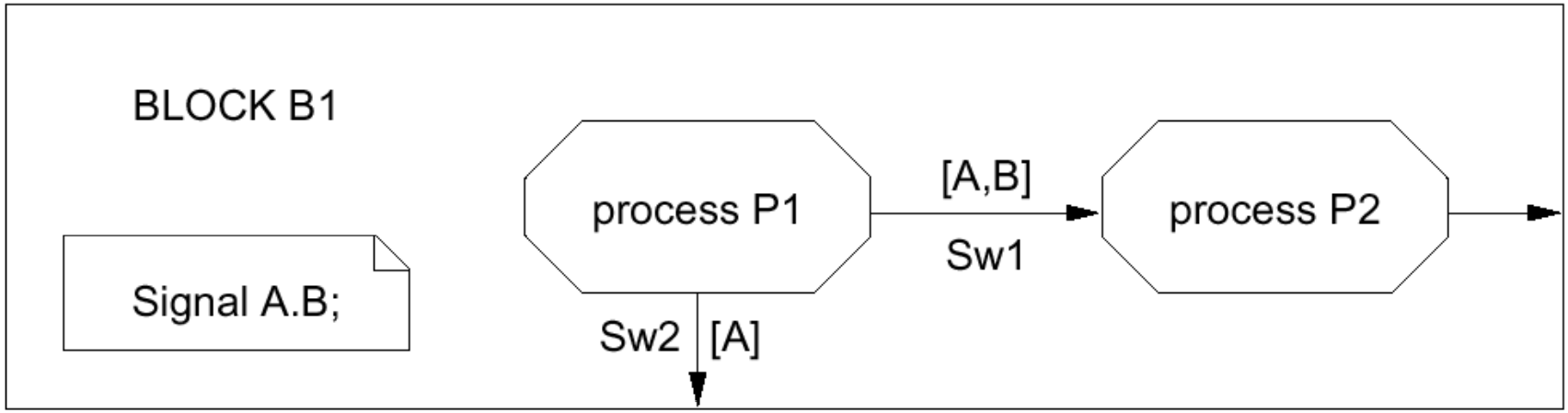
SDL : دیاگرام‌های تعامل پردازش‌ها

تعامل بین پردازش‌ها می‌تواند توسط دیاگرام‌های تعامل پردازش‌ها (حالت خاصی از دیاگرام‌های بلوکی) توصیف شود.

این دیاگرام‌ها علاوه بر پردازش‌ها حاوی کانال‌ها و توصیف سیگنال‌های محلی است.

مثال:

Process Intraction Diagram



SDL : تعیین گیرنده‌ها

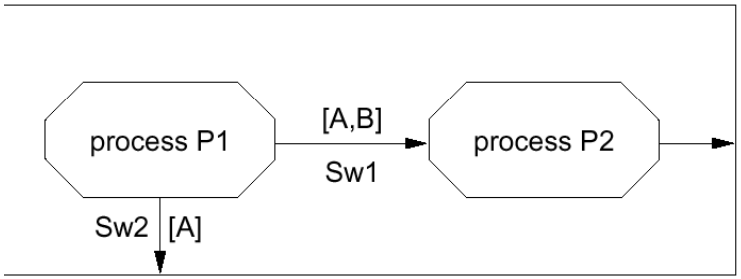
۱. **از طریق شناسه‌ی پردازش‌ها**
مثال: شناسه‌ی OFFSPRING پردازش‌هایی را بازنمایی می‌کند که به طور پویا تولید شده‌اند.

۲. **به طور صریح**
با بیان نام کانال.

۳. **به طور ضمنی**
نام سیگنال به طور ضمنی نام کانال‌ها را مشخص می‌کند (B → Sw1).

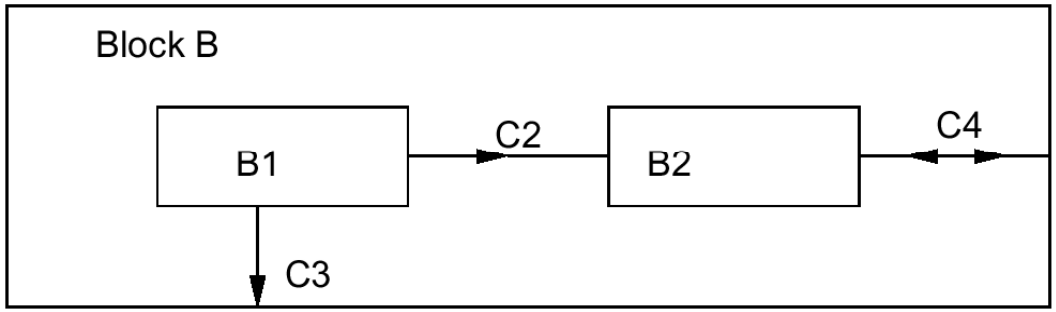
Counter
TO OFFSPRING

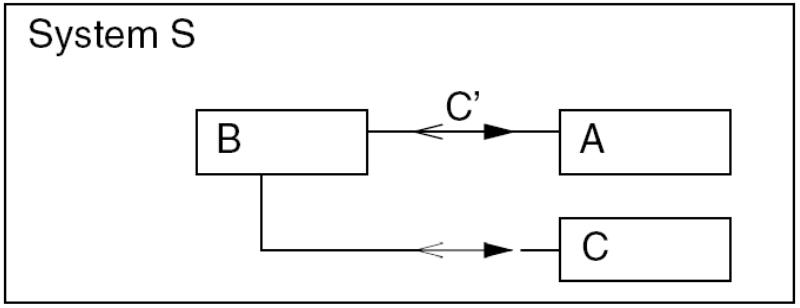
Counter
Via Sw1

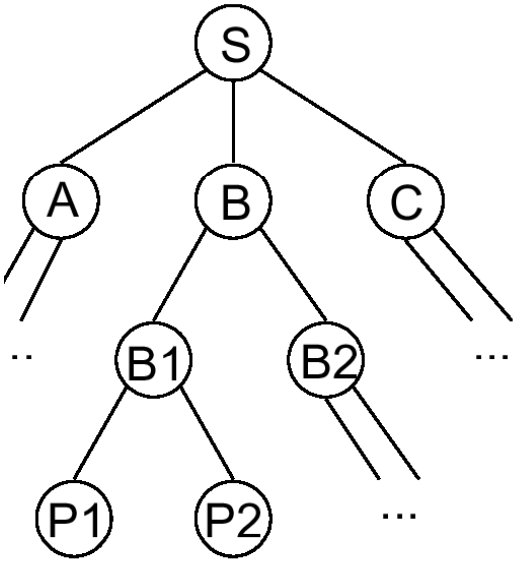


سلسله مراتب در SDL

دیاگرام‌های تعامل پردازش‌ها می‌توانند درون **بلوک‌ها** قرار گیرند.
بلوک ریشه، **سیستم** نامیده می‌شود.



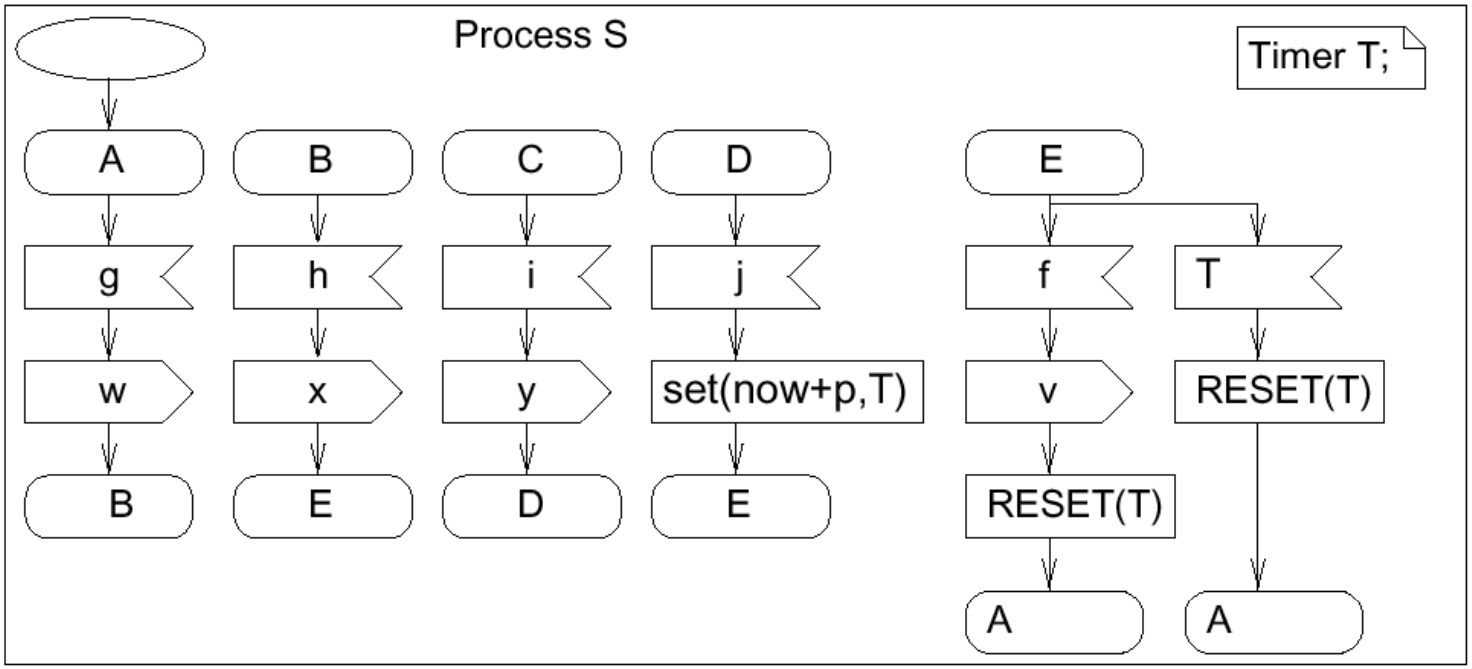




برخلاف StateChart پردازش‌ها نمی‌توانند حاوی پردازش‌های دیگر باشند.

تایمرها (زمان‌سنج‌ها)

تایمرها می‌توانند به صورت محلی تعریف شوند.
تایمرهای سپری شده، یک سیگنال را درون صف قرار می‌دهند
(لازم نیست بلافاصله پردازش شود).
RESET سیگنال تایمر را از صف حذف می‌کند.

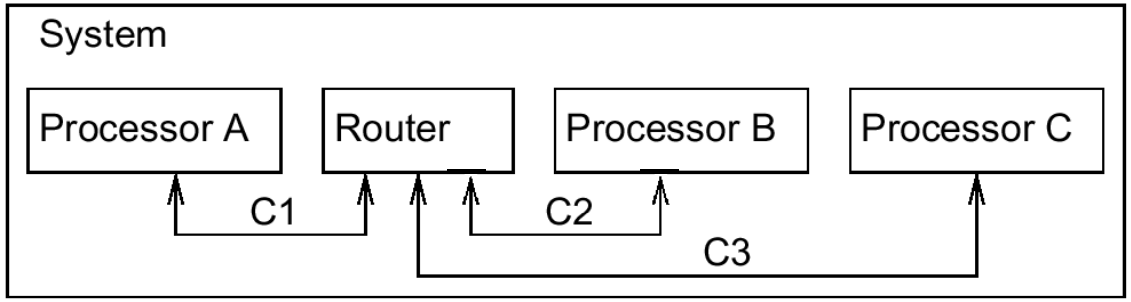


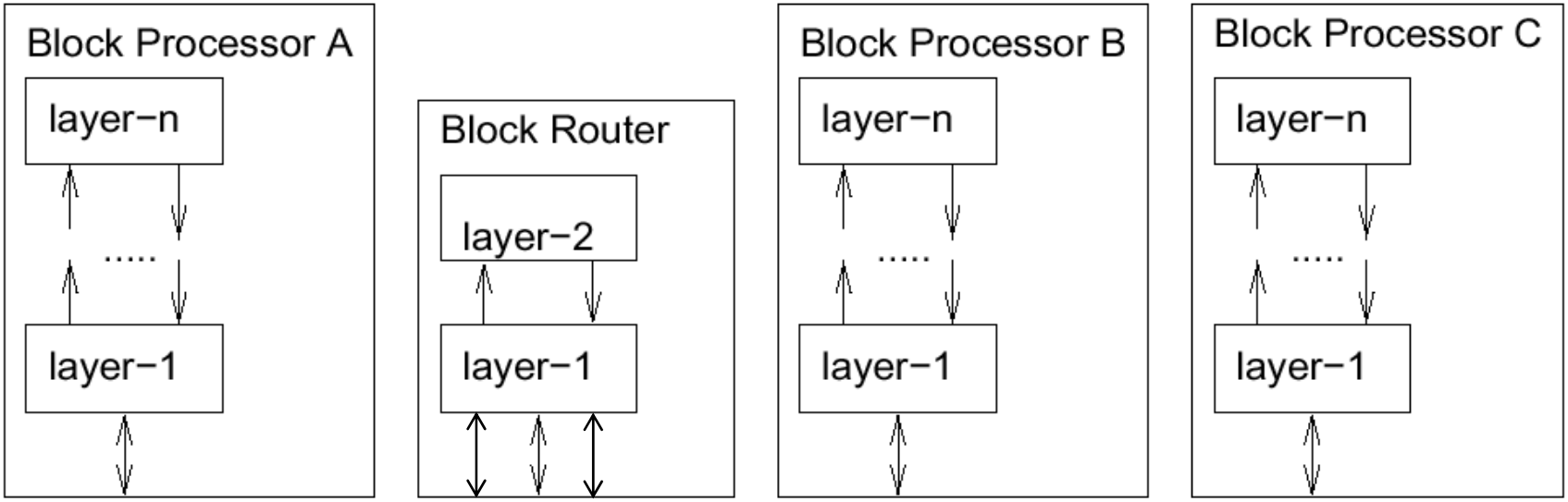
SDL : عناصر اضافی زبان

SDL حاوی چند عنصر اضافی زبان است، مانند:

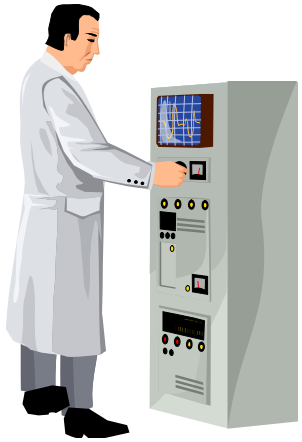
- روال‌ها
- ایجاد و خاتمه‌ی پردازش‌ها
- توصیف پیشرفته‌ی داده‌ها

کاربرد SDL: توصیف پروتکل‌های شبکه





یک مثال بزرگ‌تر از SDL: Vending Machine



این ماشین وظیفه‌ی فروش چوب‌شور، چیپس، کلوچه و نان شیرینی را بر عهده دارد:

سکه‌های پنج سنتی (nickels)، ده سنتی (dime)، ربع دلاری و نیم‌دلاری را می‌پذیرد.

این یک کاربرد توزیع‌شده نیست.

[J.M. Bergé, O. Levia, J. Roullard: High-Level System Modeling, Kluwer Academic Publishers, 1995]

System VendingMachine

Creject
[reject_coin]

Ccoins
[nickel,dime,
quarter,half]

CoinInterface

Cadd
[add]

Ccointctrl
[rej_further_coins,
accept_coins]

DecodeRequests

CamontDisplay
[amount_entered]

Cemptydisplay
[pretzel_empty, chip_empty,
cookie_empty, doughnut_empty]

CspitPurchased
[spit_pretzel, spit_chip,
spit_cookie, spit_doughnut]

Crequest
[pur_pretzel, pur_chip, pur_cookie, pur_doughnut,
reload_pretzel, reload_chip, reload_cookie,
reload_doughnut]

[spit_change] Cchange

ChangeInterface

CexaktDisplay
[exact_only]

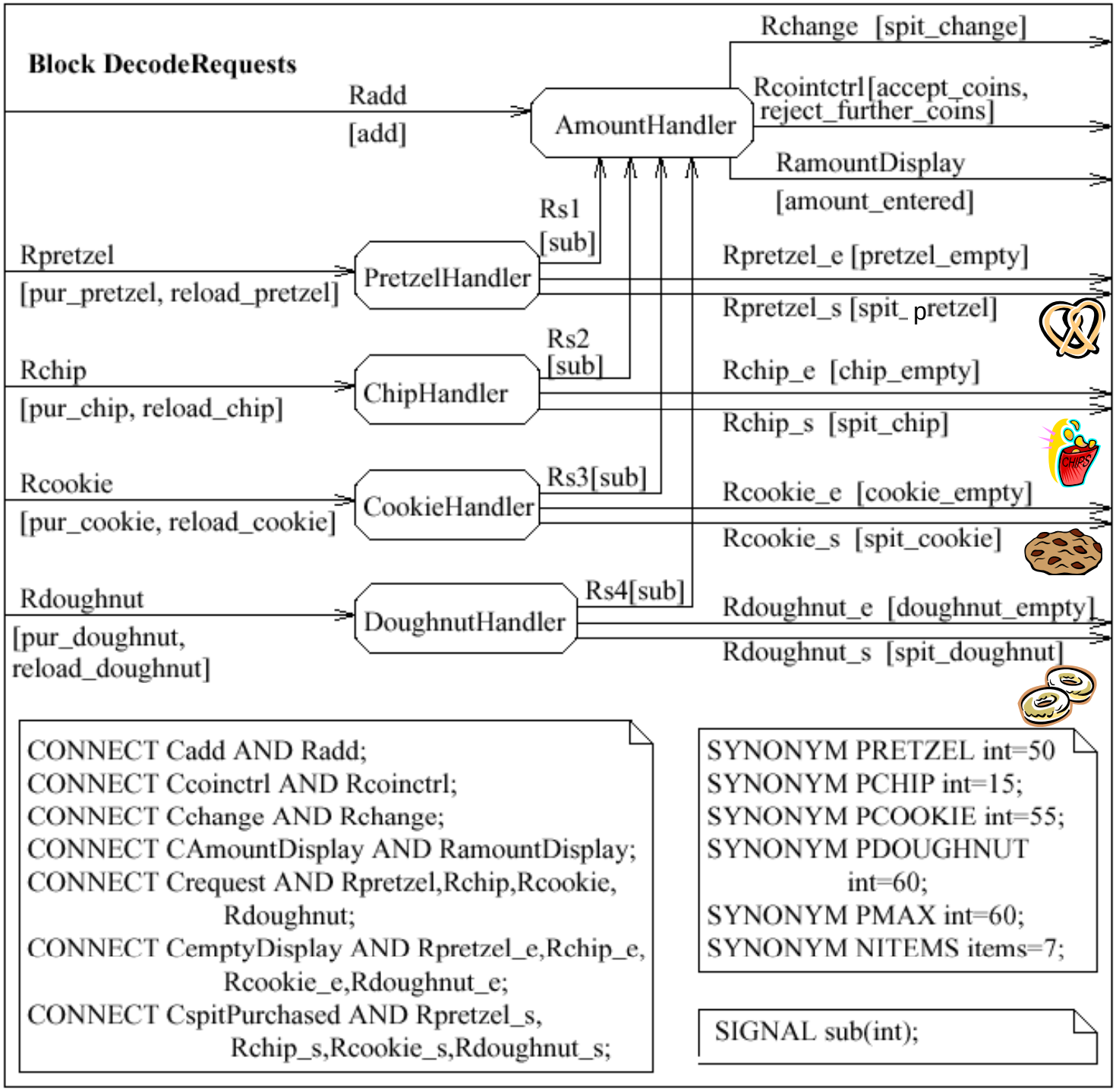
CspitChange
[spit_nickel,
spit_dime]

SIGNAL
[dime, nickel, quarter, half, pur_pretzel,
pur_cookie, pur_doughnut, pur_chip,
add(int), spit_change(int),
amount_entered(int), reject_further_coins,
exact_only, accept_coins, reject_coins,
spit_dime, spit_nickel, pretzel_empty,
spit_pretzel,chip_empty, spit_chip,
cookie_empty, spit_cookie, doughnut_empty,
spit_doughnut, reload_pretzel, reload_chip,
reload_cookie, reload_doughnut]

```
SYNTYPE items=INTEGER
  CONSTANTS 0:7
ENDSYNTYPE items;
```

```
SYNTYPE int=INTEGER
  CONSTANTS 0:127
ENDSYNTYPE int;
```

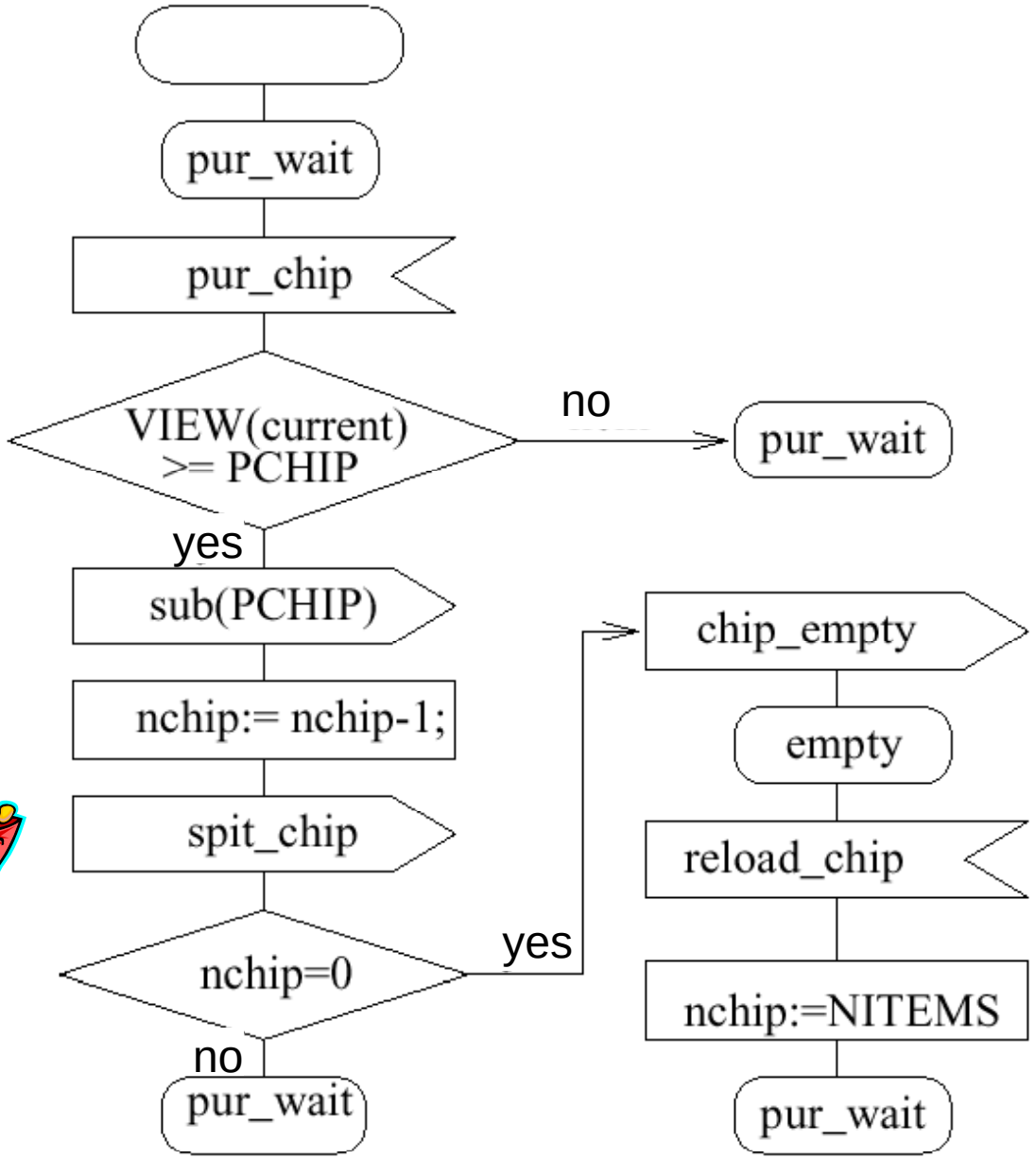
کدگشایی درخواست‌ها



Process ChipHandler

DCL nchip items:=NITEMS;

VIEWED current int;



ChipHandler

SDL: نسخه‌ها و ابزارها

- SDL-88.
- SDL-92: اضافه شدن شیئ‌گرایی
- SDL-96.
- SDL-2000: گسترش پشتیبانی گرافیکی
  پردازش‌ها و بلوک‌ها با عامل‌ها (agents) جایگزین می‌شود.
- ابزارهایی برای اتصال به MSCها
- ابزارهایی برای اتصال به UML
- ابزارهایی برای ترجمه به CHILL
- اطلاعات بیشتر در اینترنت: www.sdl-forum.org

ارزیابی SDL

- عالی برای کاربردهای توزیع‌شده (استفاده شده برای مشخص‌سازی ISDN)،
- ابزارهای تجاری موجود در SINTEF، Telelogic، Cinderella،
- لزوماً قطعی (deterministic) نیست،
  (ترتیب خواندن ورودی توسط FSMها معلوم نیست ⇐ زبان غیرهمگام)
- پیاده‌سازی آن مستلزم قرار دادن کران بر روی حداکثر طول FIFOهاست، اما محاسبه‌ی این کران می‌تواند بسیار دشوار باشد،
- مفهوم تایمر تنها برای مهلت‌های زمانی نرم کافی است (soft deadlines)،
- روش محدود شده برای به کارگیری سلسله مراتب،
- پشتیبانی محدود زبان‌های برنامه‌سازی،
- توصیف ویژگی‌های غیرتابعی امکان ندارد.

خلاصه

- **General language properties**
  - Synchronous vs. asynchronous languages
  - Properties of processes
  - Communication
  - Timing
  - Access to special I/O devices
- **SDL**
  - Representation of processes
  - Communication & block diagrams
  - Timers and other language elements
  - Example: Vending machine
  - Versions and evaluation