IRMIS Universal Component Types

D. A. Dohan
NSLS 2
Brookhaven National Laboratory

ICALEPCS 2007

Universal Component Types

Introduction

In an operating facility, the best (and usually the only) indicator of failed hardware is *software* – an EDM screen, an alarm handler, script, etc.

→ Goal: relate the accelerator software to the hardware ←

- - what do we mean by ‘hardware’?

- - is it possible to have a common definition of ‘hardware’?

Universal Component Types

The IRMIS component model has been successful in capturing the APS control system hardware and infrastructure. This past year has been a period of data entry, validation, consolidation and collecting new ideas for IRMIS. The component database is now complete, and contains:

- > 30,000 components
- ~ 900 component-types
- primary mechanism for trouble shooting control system problems
- rich set of applications that use the component database (AOI, idt, gst, CCMS, ioc,…)

The IRMIS component model contains/provides

- site-neutral component-type definitions
- exhaustive coverage

Universal Component Types

At NSLS2, we want to leverage off this effort, and to extend the component-type definitions to include:

- beam delivery components (magnets, bpms, …) – *support for machine modeling*
- power supplies
- beam line components
- IT
- plant facilities
- . . .

Universal Component Types

What is an IRMIS component-type?

- IRMIS components are common, familiar infrastructure elements (rooms, racks), COTS items (chassis, processors, I/O boards), instruments, …

- derived by successive system partitioning, until the ‘unit-replaceable’ point is reached. This promotes exhaustive coverage, which is difficult to achieve using a modeling approach.

- site independent component type definitions – everyone builds accelerators using the same set of vendor catalogues

- the concept of component types began with the idea of EPICS supported devices, and has been generalized to include all infrastructure items needed to operate the facility

Universal Component Types

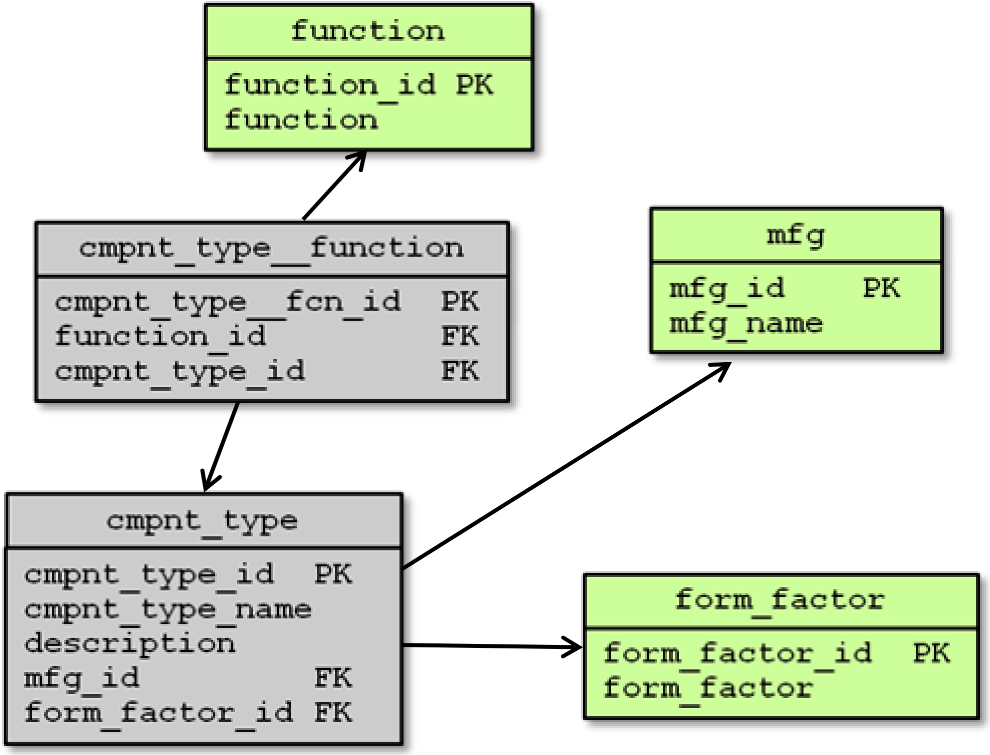
Component are <u>not</u> objects, classes or devices

- no ‘role’ is assigned to components. In IRMIS, a magnet is not treated as something that is a corrector or a bend – it is simply a component whose function it is to convert an electric current into a magnetic field.

- no role-based table names – e.g. linac-xxx, etc. Again, this would make the RDB site-specific.

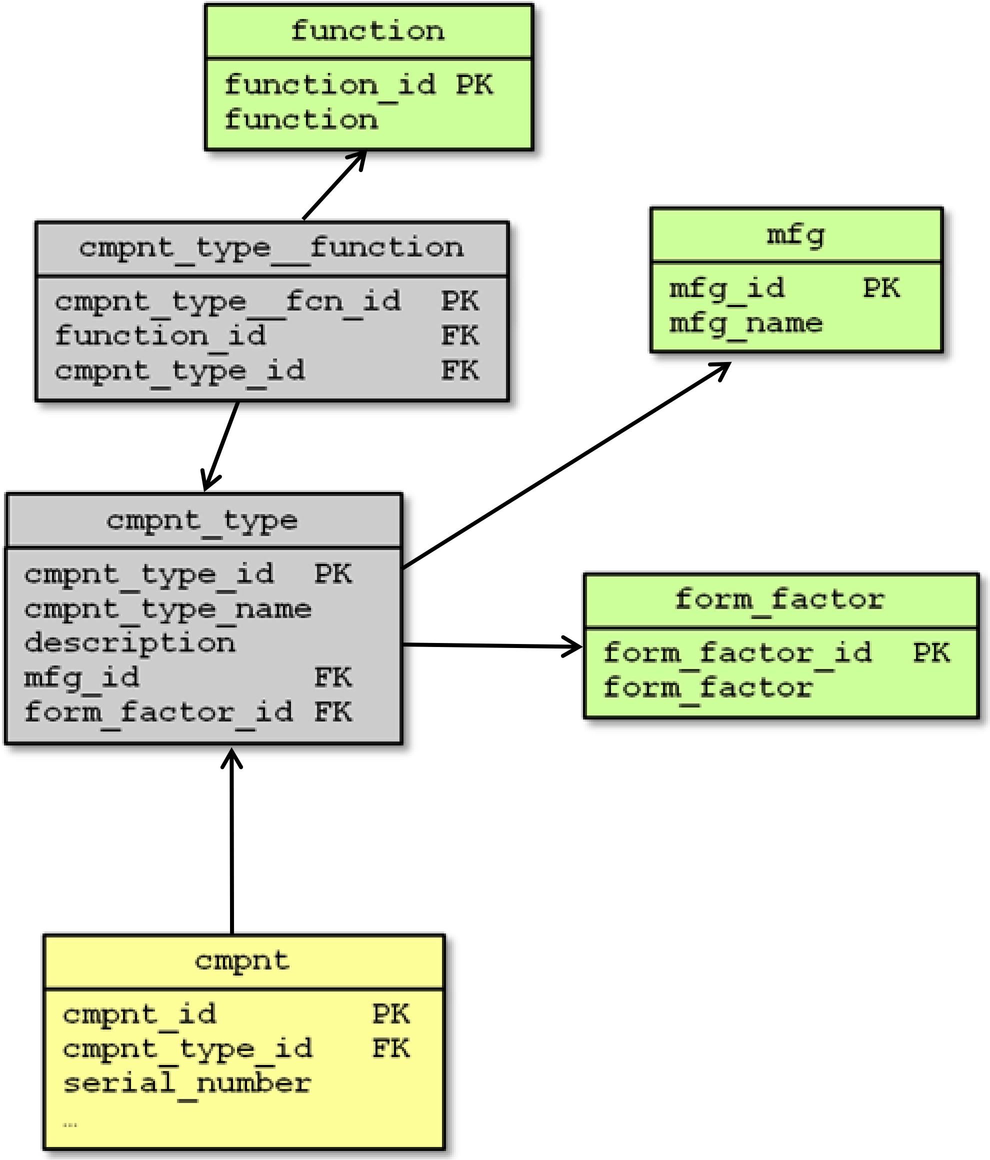
- no behavior, methods, etc are assigned to components

What is an IRMIS Component-Type?



```
function
function_id PK
function

cmpnt_type__function
cmpnt_type__fcn_id   PK
function_id          FK
cmpnt_type_id        FK

mfg
mfg_id      PK
mfg_name

cmpnt_type
cmpnt_type_id   PK
cmpnt_type_name
description
mfg_id          FK
form_factor_id  FK

form_factor
form_factor_id   PK
form_factor
```

What is an IRMIS Component-Type?



function	
function_id	PK
function	

cmpnt_type__function	
cmpnt_type__fcn_id	PK
function_id	FK
cmpnt_type_id	FK

mfg	
mfg_id	PK
mfg_name	

cmpnt_type	
cmpnt_type_id	PK
cmpnt_type_name	
description	
mfg_id	FK
form_factor_id	FK

form_factor	
form_factor_id	PK
form_factor	

cmpnt	
cmpnt_id	PK
cmpnt_type_id	FK
serial_number	
…	

3-Hierarchy Component Model

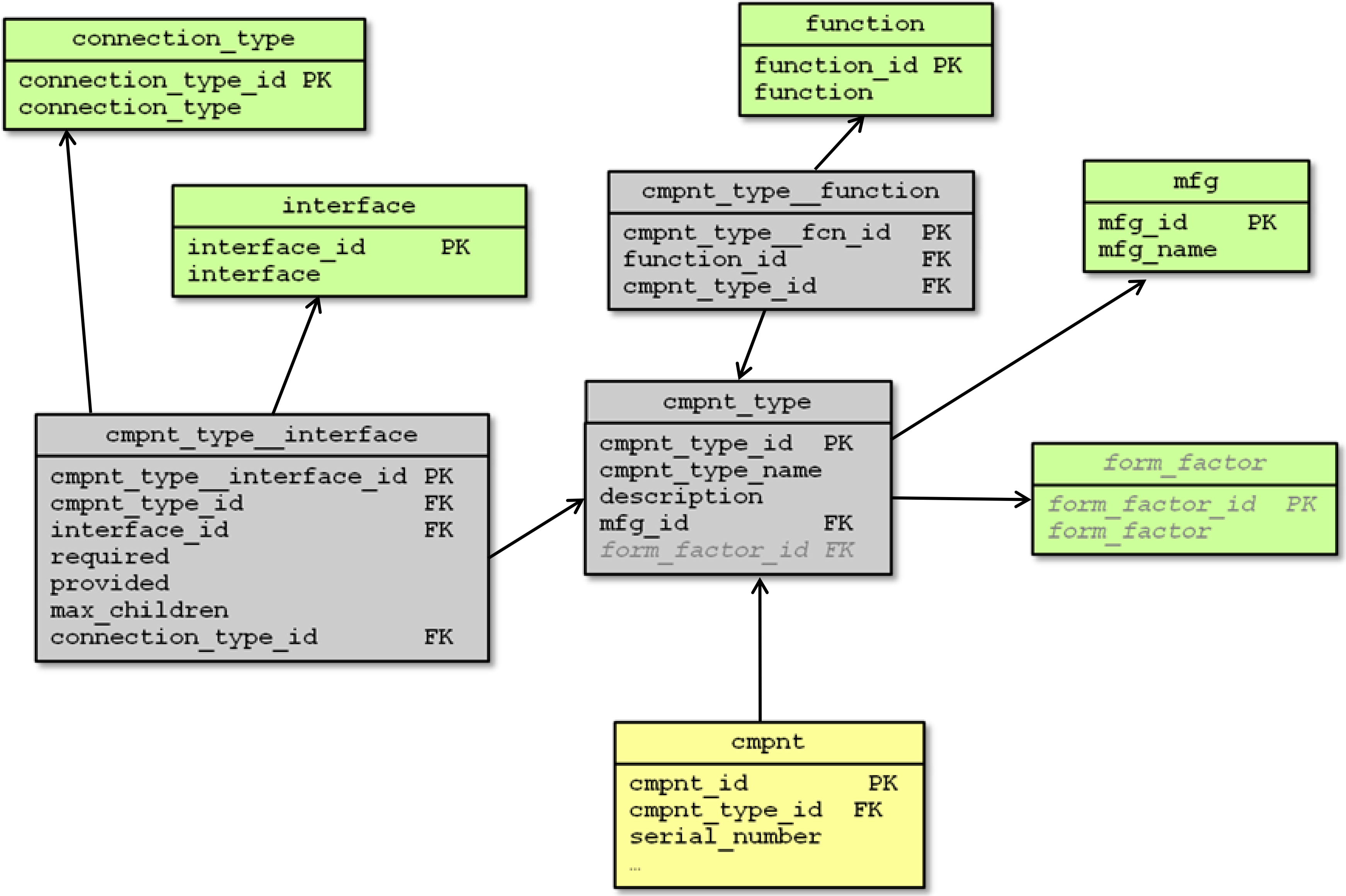
Hierarchical component assembly:

Housing: Every component is a member of the housing hierarchy --> every component is housed somewhere, thus has a unique housing parent (which is another component somewhere in the system)

Control: Each controlled component has a unique control parent, implying a hierarchy leading ultimately to the IOC or control 'root' for the component

Power: This hierarchy captures the power distribution grid. This allows the power source for any active component to be traced to the originating switch gear.

What is an IRMIS Component-Type?



connection_type

connection_type_id PK

connection_type

function

function_id PK

function

interface

interface_id PK

interface

cmpnt_type__function

cmpnt_type__fcn_id PK

function_id FK

cmpnt_type_id FK

mfg

mfg_id PK

mfg_name

cmpnt_type__interface

cmpnt_type__interface_id PK

cmpnt_type_id FK

interface_id FK

required

provided

max_children

connection_type_id FK

cmpnt_type

cmpnt_type_id PK

cmpnt_type_name

description

mfg_id FK

form_factor_id FK

form_factor

form_factor_id PK

form_factor

cmpnt

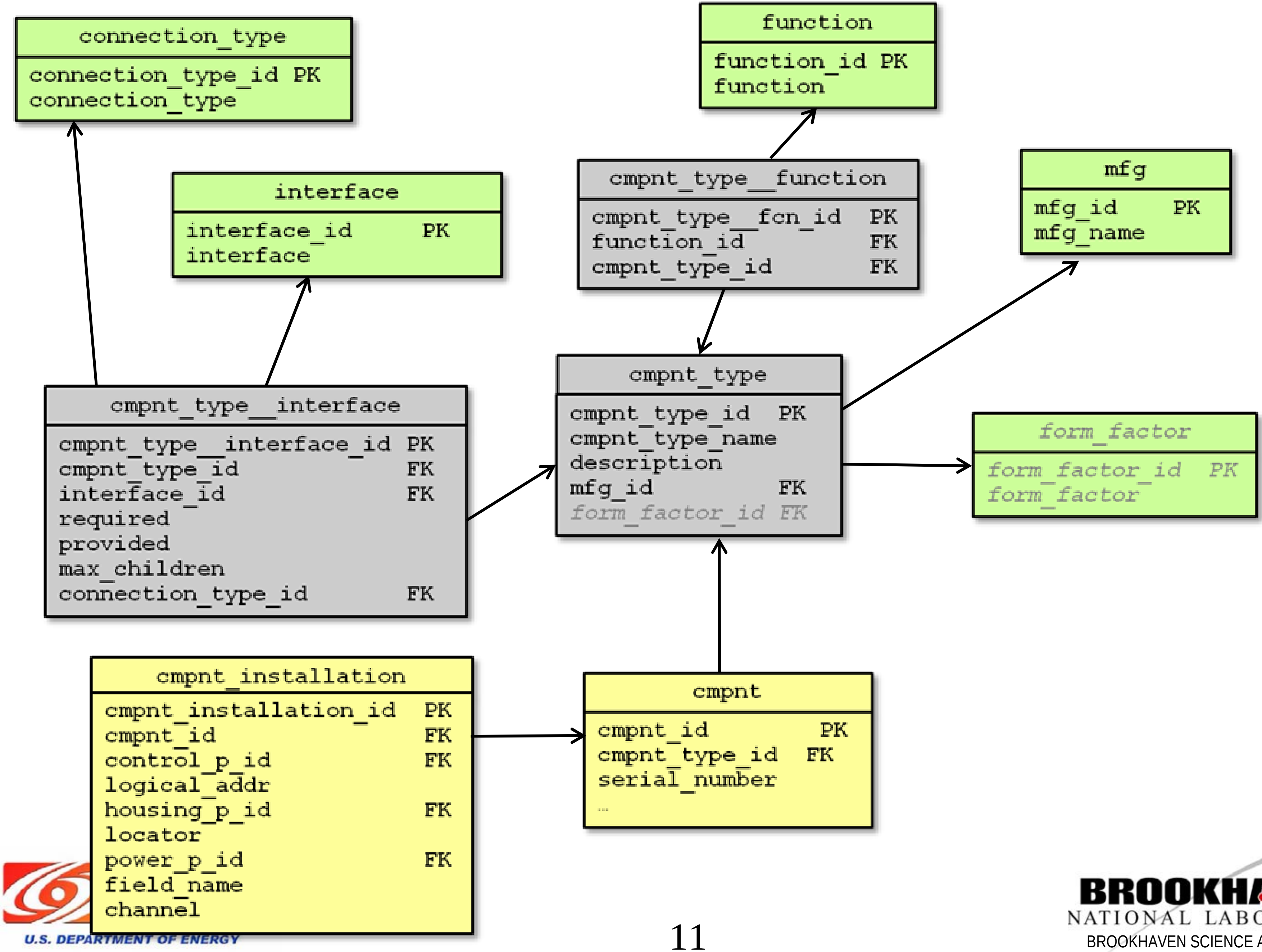
cmpnt_id PK

cmpnt_type_id FK

serial_number

...

What is an IRMIS Component-Type?



```
connection_type
connection_type_id PK
connection_type
```

```
interface
interface_id       PK
interface
```

```
function
function_id PK
function
```

```
cmpnt_type__function
cmpnt_type__fcn_id   PK
function_id          FK
cmpnt_type_id        FK
```

```
mfg
mfg_id      PK
mfg_name
```

```
cmpnt_type__interface
cmpnt_type__interface_id PK
cmpnt_type_id            FK
interface_id             FK
required
provided
max_children
connection_type_id       FK
```

```
cmpnt_type
cmpnt_type_id   PK
cmpnt_type_name
description
mfg_id          FK
form_factor_id  FK
```

```
form_factor
form_factor_id   PK
form_factor
```

```
cmpnt_installation
cmpnt_installation_id PK
cmpnt_id              FK
control_p_id          FK
logical_addr
housing_p_id          FK
locator
power_p_id            FK
field_name
channel
```

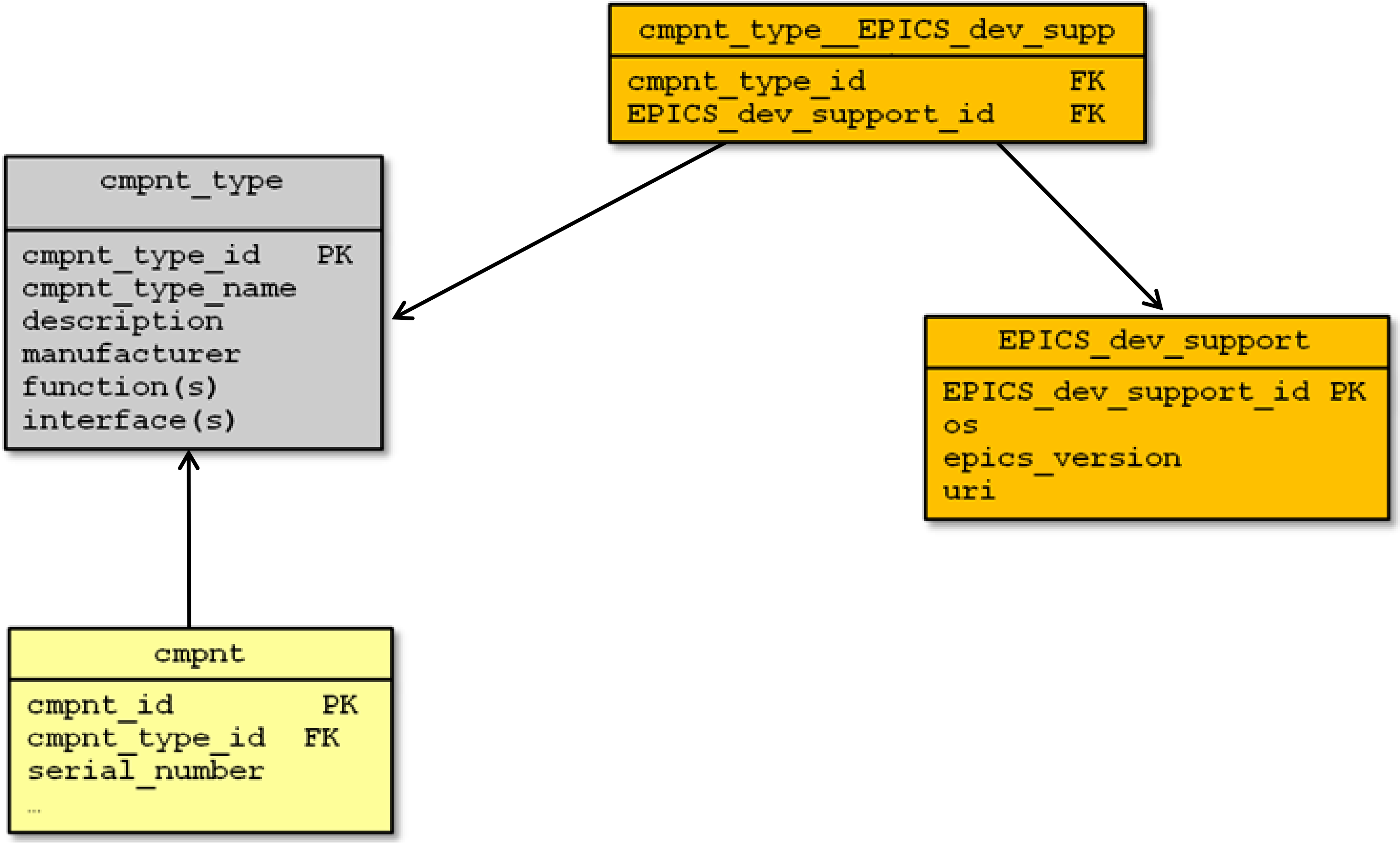
```
cmpnt
cmpnt_id          PK
cmpnt_type_id    FK
serial_number
…
```

Finding existing software

- **Supported Hardware database on EPICS website**
- **EPICS Collaboration meetings**
- **Tech-talk mailing list**
  - Search the archives before asking
- **Google Search**

- **Most existing support software is for R3.13.x on vxWorks**
- **Porting to R3.14 is relatively easy**
- **Porting to RTEMS is slightly harder**
  - Should be straight-forward if the software uses devLib
- **Porting to Linux is inadvisable for VME or PCI-based hardware**
  - Linux Kernel drivers are hard to write and debug
- **Serial or network-based support can be made as portable as Base**
  - The Asyn framework simplifies this process

Component Types – Wish List



cmpnt_type__EPICS_dev_supp

cmpnt_type_id FK
EPICS_dev_support_id FK

cmpnt_type

cmpnt_type_id PK
cmpnt_type_name
description
manufacturer
function(s)
interface(s)

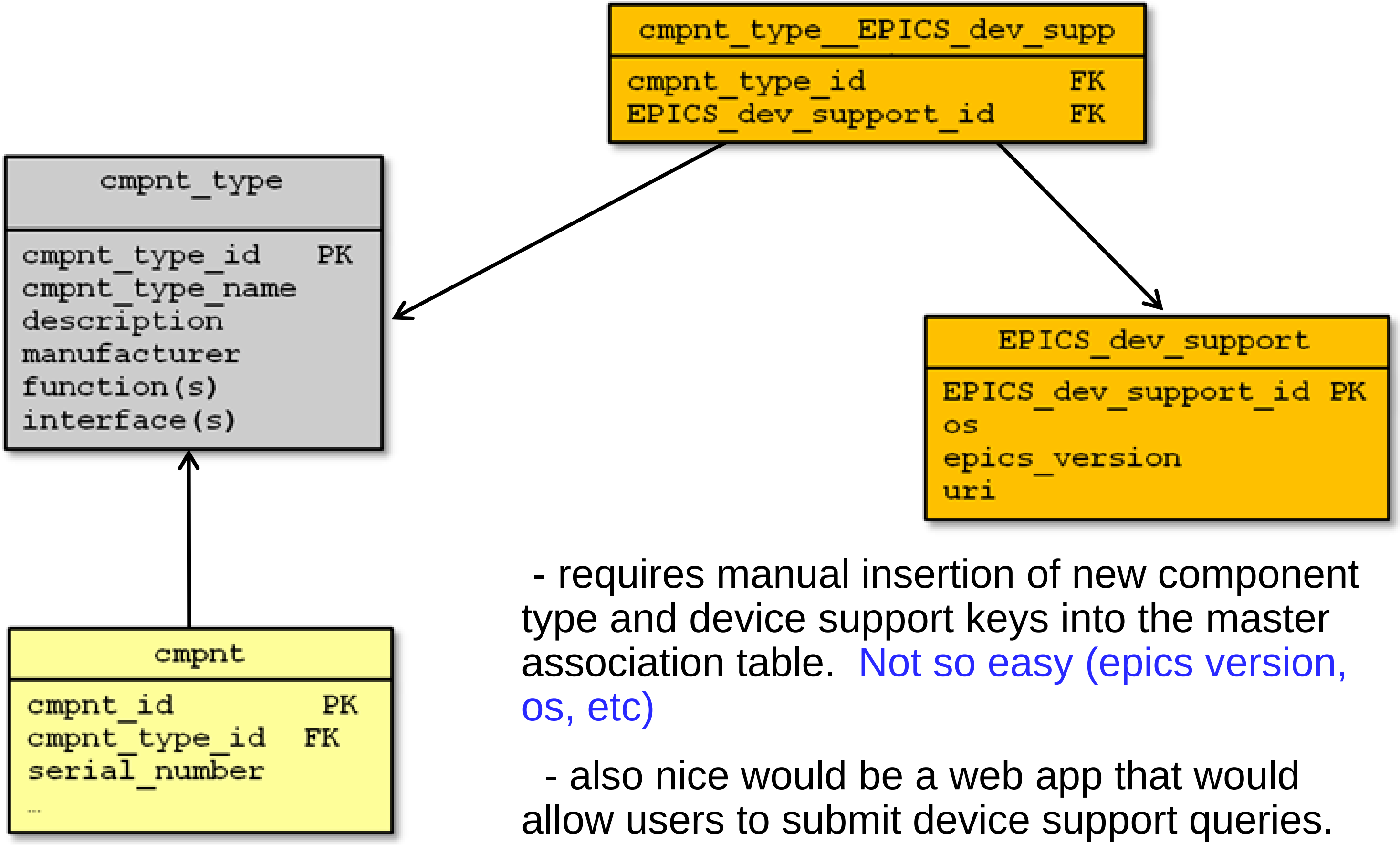
EPICS_dev_support

EPICS_dev_support_id PK
os
epics_version
uri

cmpnt

cmpnt_id PK
cmpnt_type_id FK
serial_number
…

Component Types – Wish List



cmpnt_type__EPICS_dev_supp

cmpnt_type_id FK
EPICS_dev_support_id FK

cmpnt_type

cmpnt_type_id PK
cmpnt_type_name
description
manufacturer
function(s)
interface(s)

EPICS_dev_support

EPICS_dev_support_id PK
os
epics_version
uri

cmpnt

cmpnt_id PK
cmpnt_type_id FK
serial_number
...

- requires manual insertion of new component type and device support keys into the master association table. Not so easy (epics version, os, etc)

- also nice would be a web app that would allow users to submit device support queries.

Sharing Component Types

Sharing Component Type Definitions

- to model an installed system using the IRMIS component model, you really want to leverage off the APS investment in developing and refining the rich component-type relational schema.

- implies a central repository (does not need to be a shareable relational database - can be a flat, csv file from which the local site could generate local RDB)

- a mechanism is needed whereby the component-type RDB table can be extended from any site (including the APS)

– to prevent a divergence in the component type definitions

Component Types - Extended

The IRMIS PV schema is populated by a non-invasive crawler. It provides a global view of the installed EPICS real-time software, inter-IOC/inter-record links, and EPICS client application references to PVs. Widely adopted schema, used in multiple laboratories

Attempts to mine the EPICS PV database (dbhcr reports, DTYPS, etc) for hardware information have had limited success. This heuristic approach was used to populate the original APS component database, but only for the controls hierarchy. Housing and power hierarchies were populated with another set of heuristics, also error prone. The rest was done by hand.

With the component database in hand --

Component Types - Extended

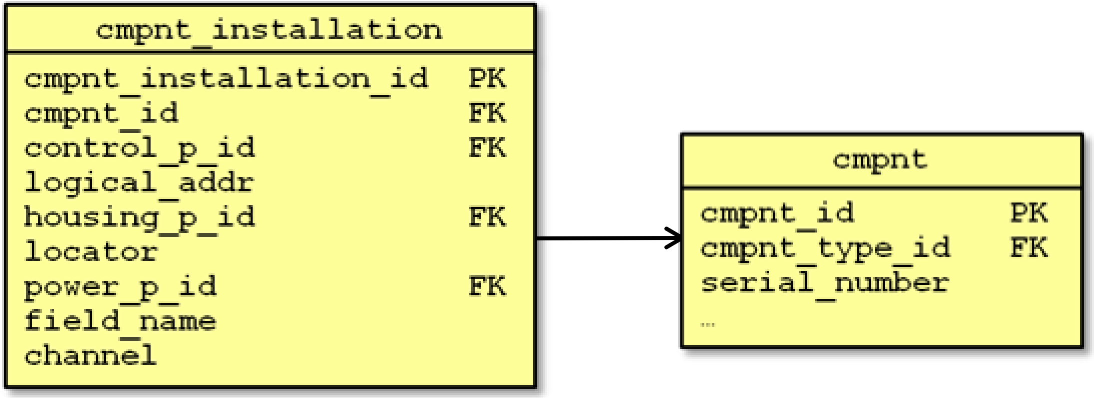
The IRMIS PV schema is populated by a non-invasive crawler. It provides a global view of the installed EPICS real-time software, inter-IOC/inter-record links, and EPICS client application references to PVs. Widely adopted schema, used in multiple laboratories

Attempts to mine the EPICS PV database (dbhcr reports, DTYPS, etc) for hardware information have had limited success. This heuristic approach was used to populate the original APS component database, but only for the controls hierarchy. Housing and power hierarchies were populated with another set of heuristics, also error prone. The rest was done by hand.

With the component database in hand --

→ how do we relate the accelerator software to the hardware? ←

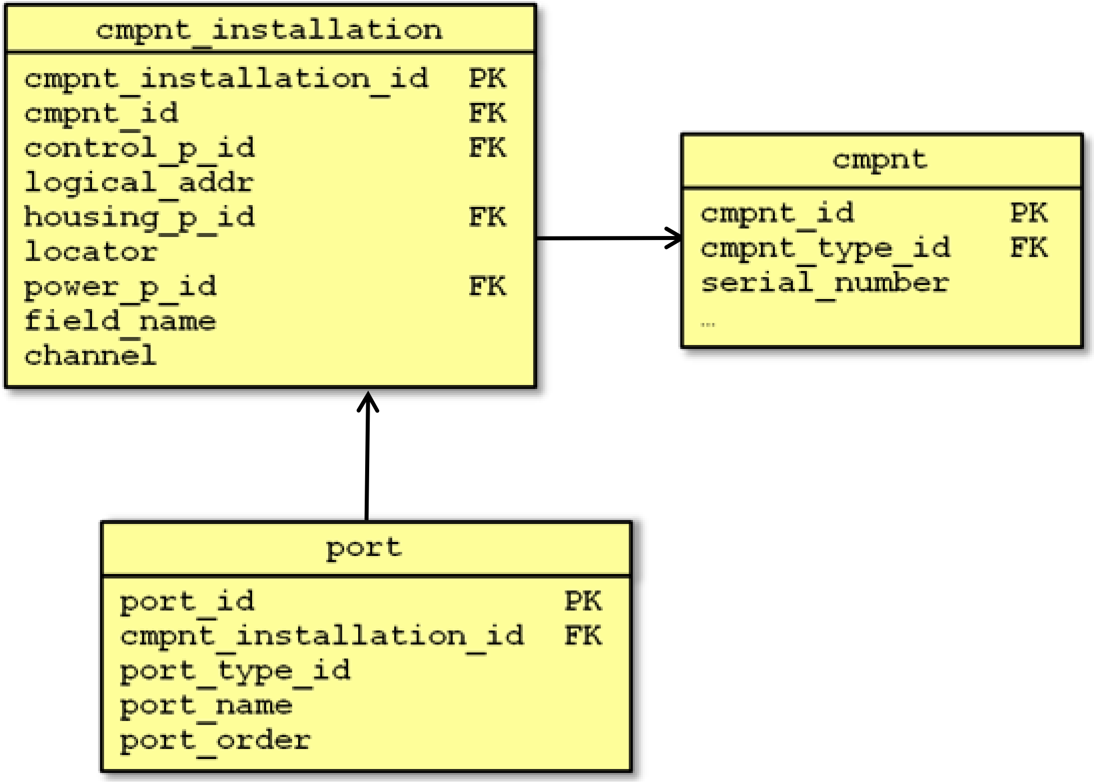
Component Types - Extended



cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

cmpnt	
cmpnt_id	PK
cmpnt_type_id	FK
serial_number	
…	

Component Types - Extended

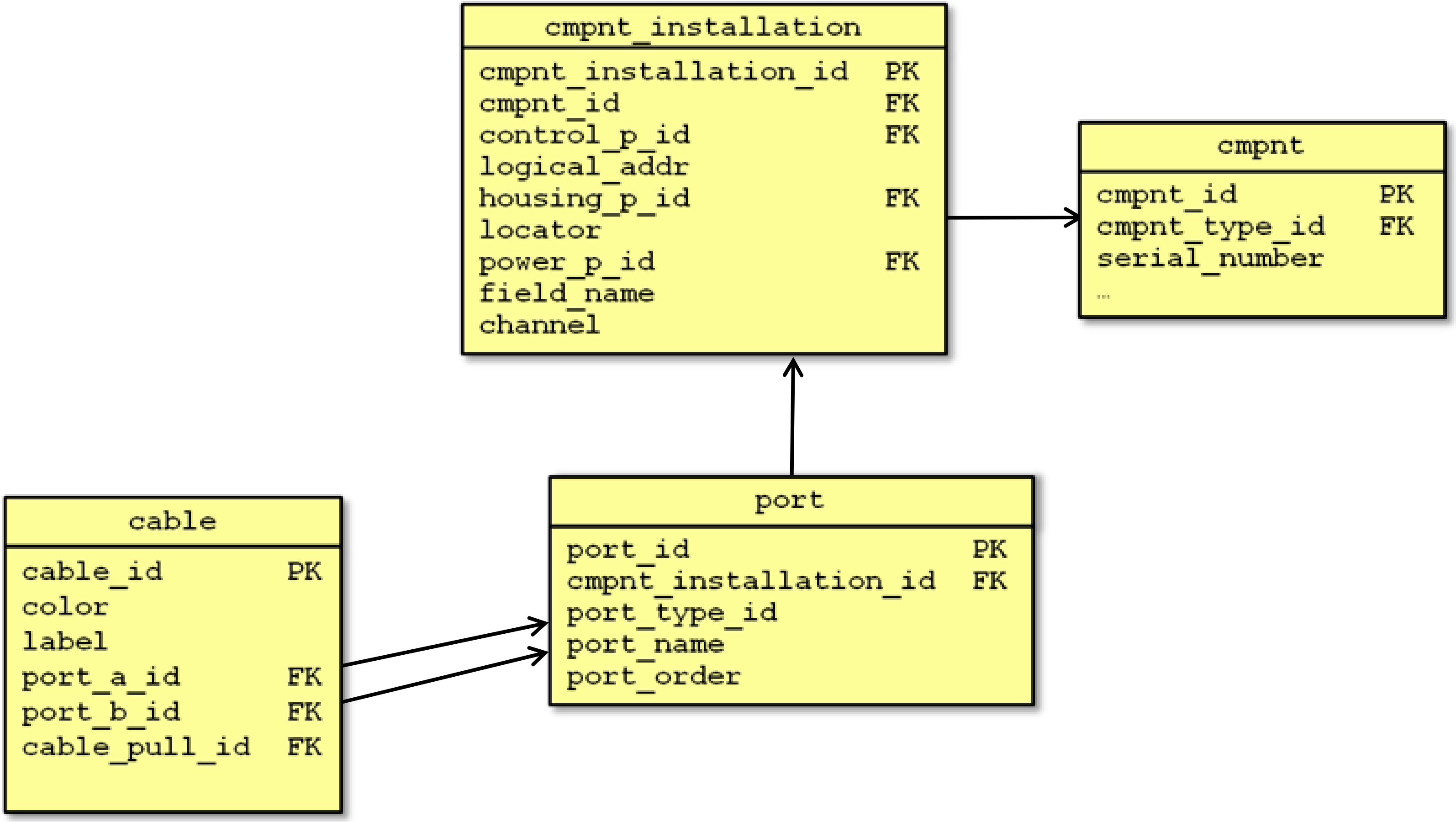


cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

cmpnt	
cmpnt_id	PK
cmpnt_type_id	FK
serial_number	
...	

port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

Component Types - Extended



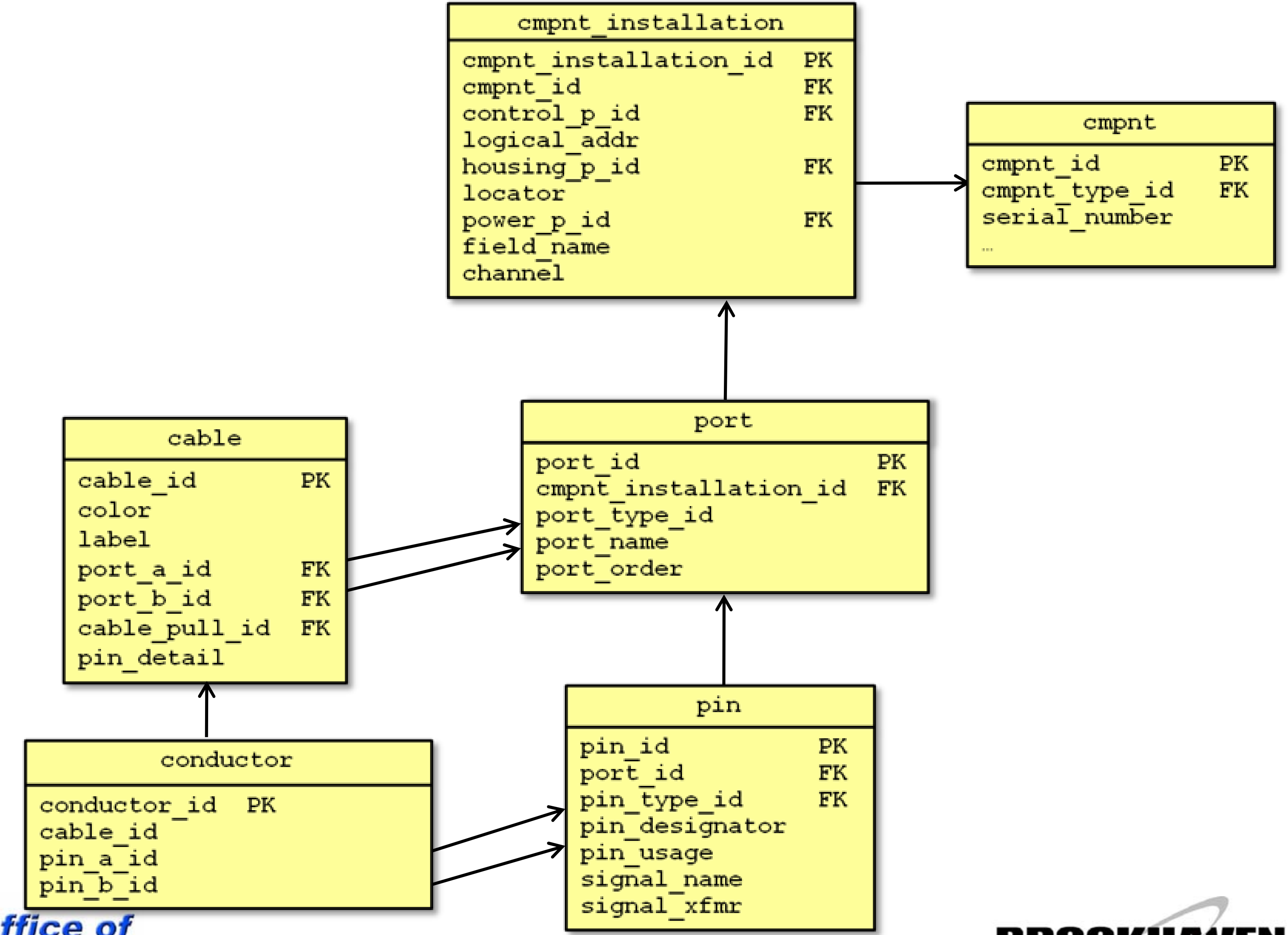
cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

cmpnt	
cmpnt_id	PK
cmpnt_type_id	FK
serial_number	
…	

cable	
cable_id	PK
color	
label	
port_a_id	FK
port_b_id	FK
cable_pull_id	FK

port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

Component Types - Extended



cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

cmpnt	
cmpnt_id	PK
cmpnt_type_id	FK
serial_number	
…	

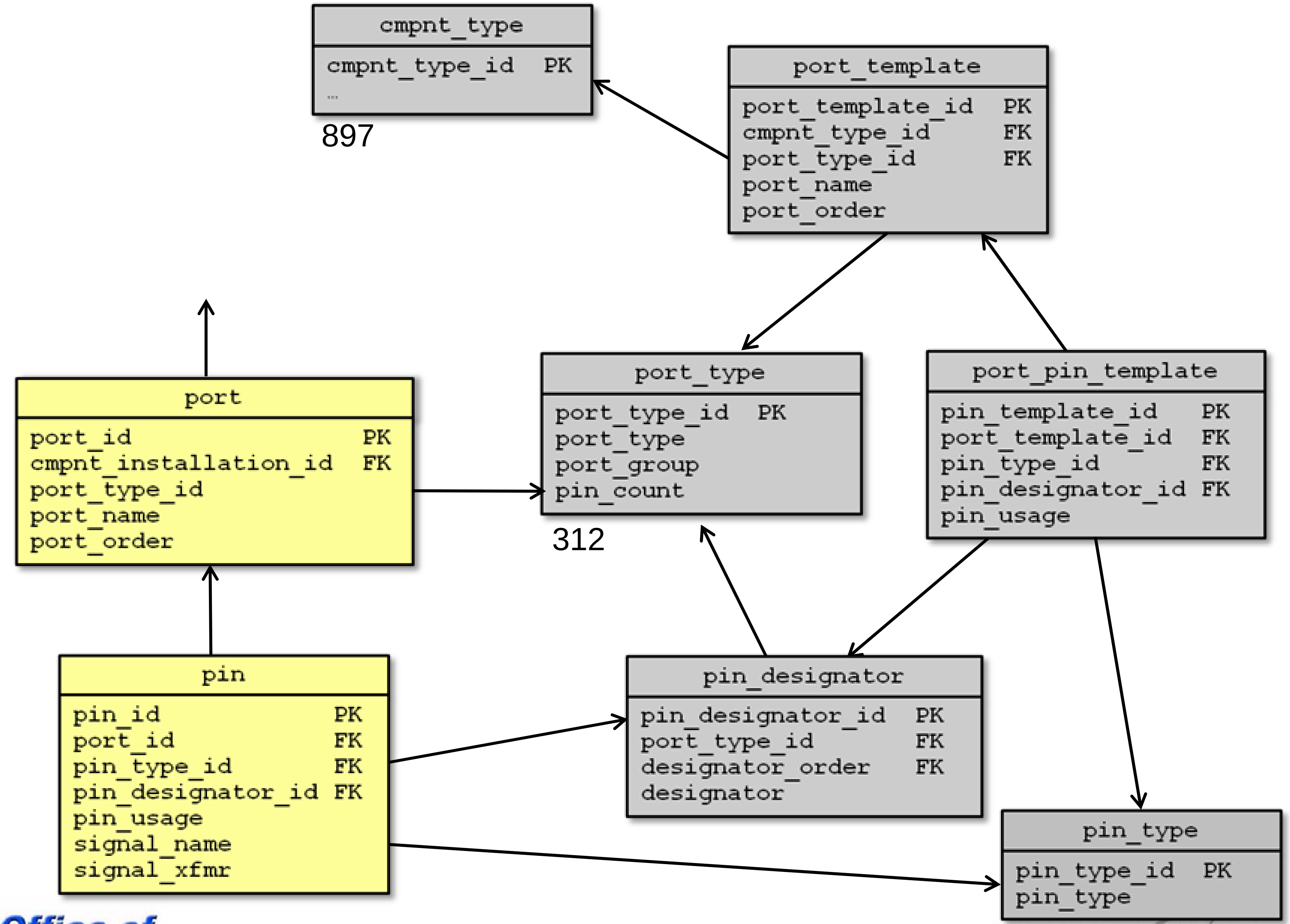
port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

cable	
cable_id	PK
color	
label	
port_a_id	FK
port_b_id	FK
cable_pull_id	FK
pin_detail	

pin	
pin_id	PK
port_id	FK
pin_type_id	FK
pin_designator	
pin_usage	
signal_name	
signal_xfmr	

conductor	
conductor_id	PK
cable_id	
pin_a_id	
pin_b_id	

Component Types - Extended



cmpnt_type
cmpnt_type_id PK
...

897

port_template
port_template_id PK
cmpnt_type_id FK
port_type_id FK
port_name
port_order

port
port_id PK
cmpnt_installation_id FK
port_type_id
port_name
port_order

port_type
port_type_id PK
port_type
port_group
pin_count

312

port_pin_template
pin_template_id PK
port_template_id FK
pin_type_id FK
pin_designator_id FK
pin_usage

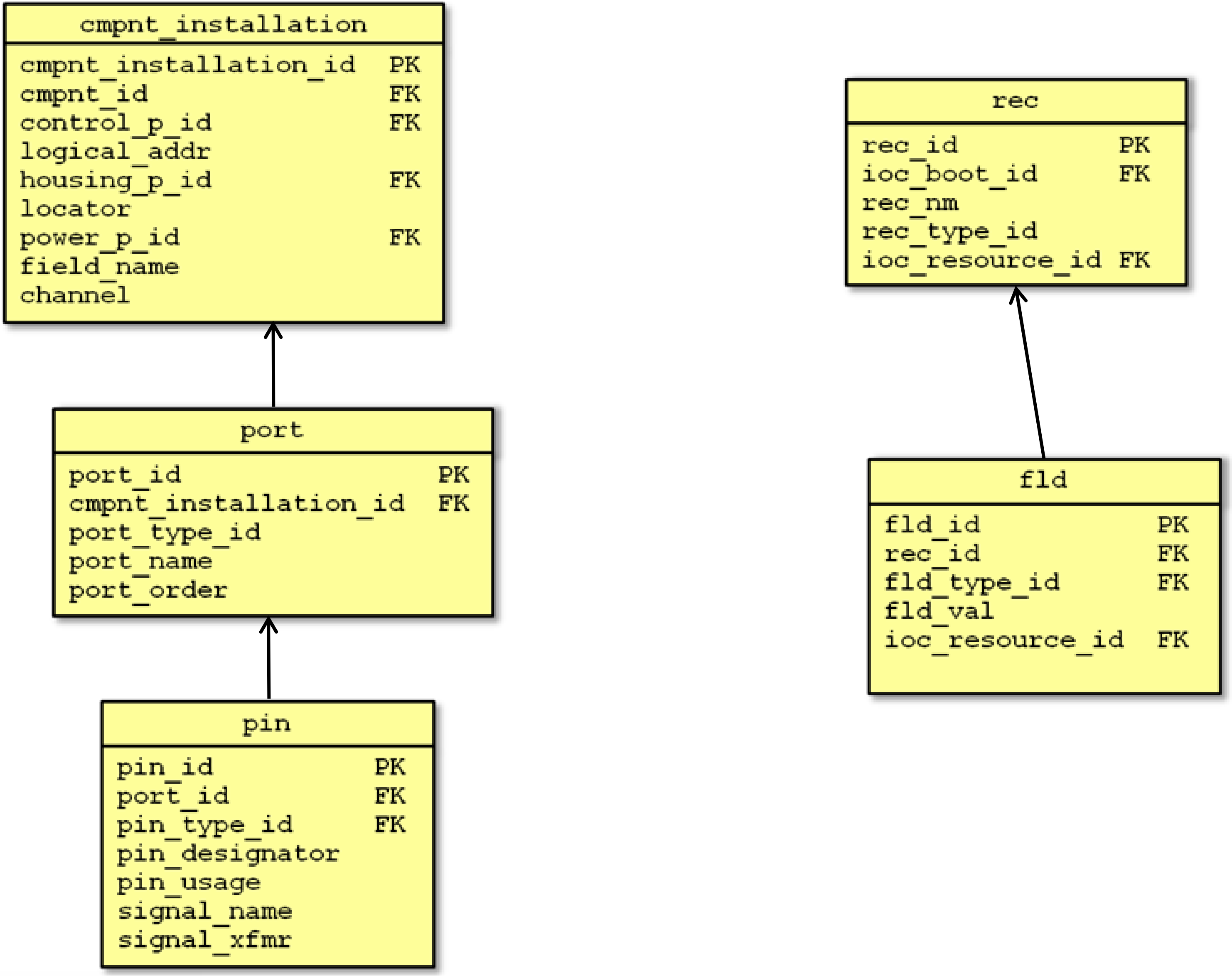
pin
pin_id PK
port_id FK
pin_type_id FK
pin_designator_id FK
pin_usage
signal_name
signal_xfmr

pin_designator
pin_designator_id PK
port_type_id FK
designator_order FK
designator

pin_type
pin_type_id PK
pin_type

→ how do we relate the accelerator software to the hardware? ←

Component – PV relationships



cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

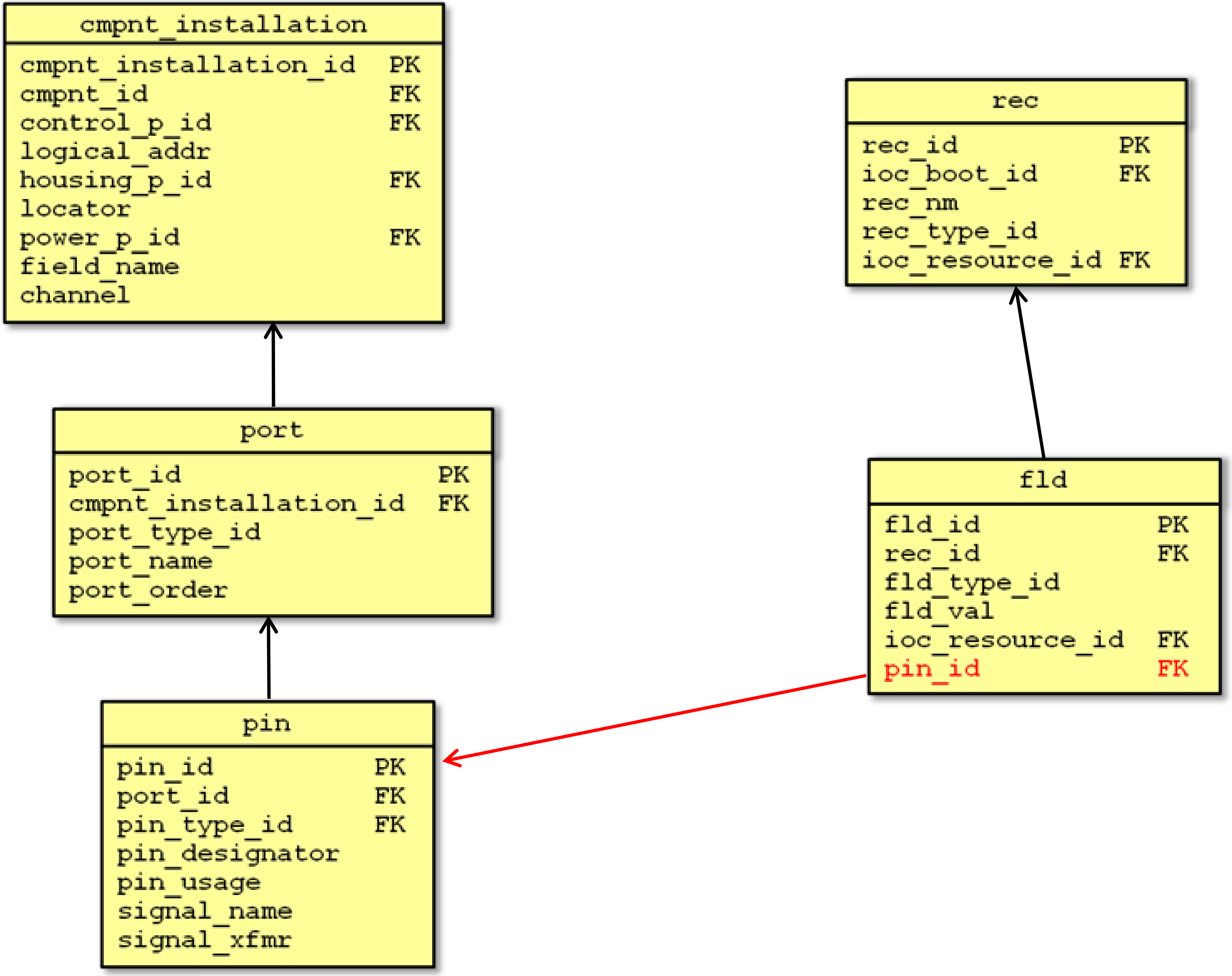
port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

pin	
pin_id	PK
port_id	FK
pin_type_id	FK
pin_designator	
pin_usage	
signal_name	
signal_xfmr	

rec	
rec_id	PK
ioc_boot_id	FK
rec_nm	
rec_type_id	
ioc_resource_id	FK

fld	
fld_id	PK
rec_id	FK
fld_type_id	FK
fld_val	
ioc_resource_id	FK

Component – PV relationships (proposal)



cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

pin	
pin_id	PK
port_id	FK
pin_type_id	FK
pin_designator	
pin_usage	
signal_name	
signal_xfmr	

rec	
rec_id	PK
ioc_boot_id	FK
rec_nm	
rec_type_id	
ioc_resource_id	FK

fld	
fld_id	PK
rec_id	FK
fld_type_id	
fld_val	
ioc_resource_id	FK
pin_id	FK

Component – PV relationships
(proposal)

Proposal: **pv <> pin** link entered as .db comment *(similar to vdct graphical positioning comment entry)*

<u>Work flow (new facility):</u>

- user 'installs' the component into IRMIS. RDB assigns each pin a primary key

- EPICS database edit – user must be provided with the ability to drill down to the component pin-out. This primary key is then inserted into .db file.

> This could be a plug-in/extension to vdct, or companion gui with cut and paste. (needed only for I/O records)

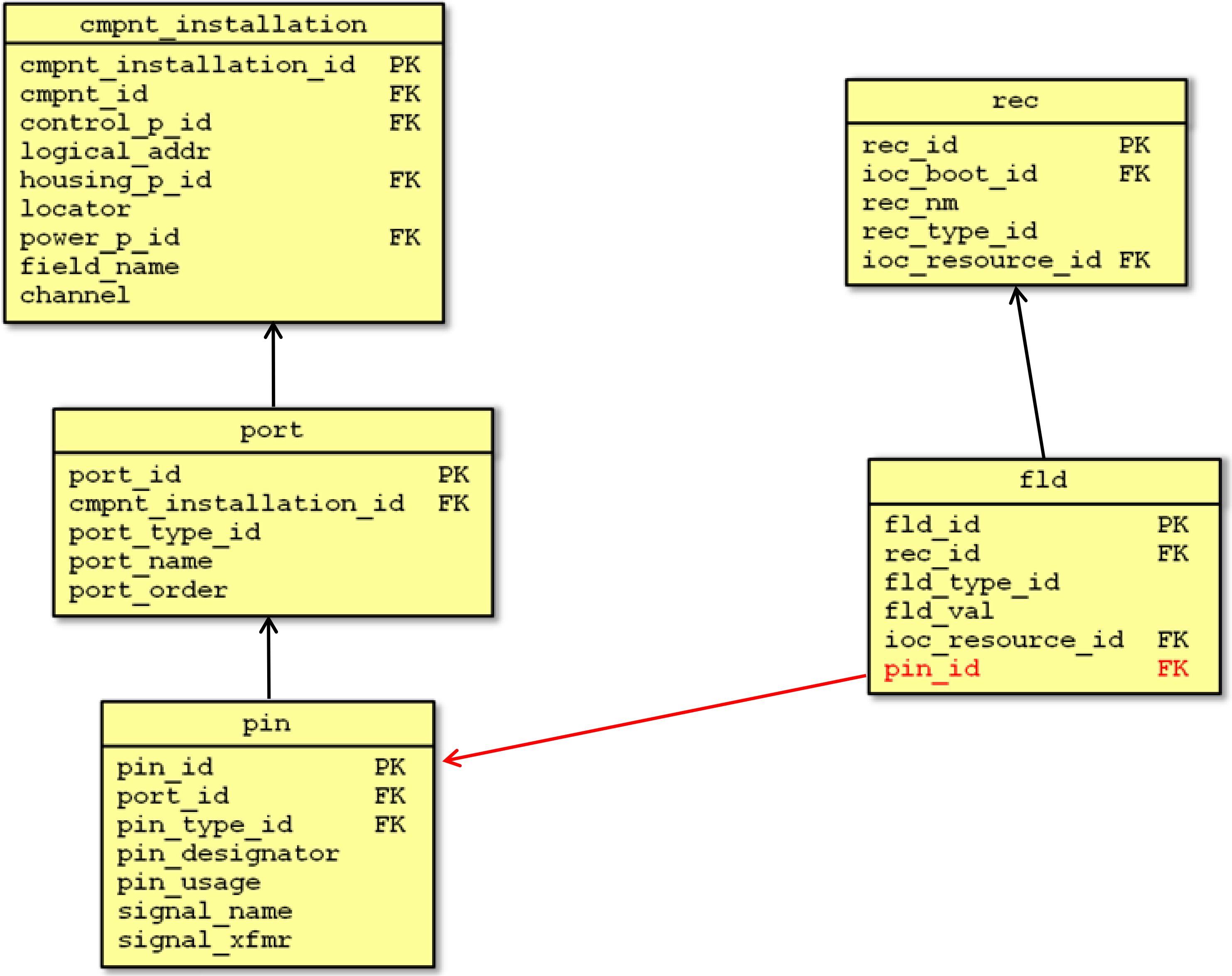
Component – PV relationships
(proposal)

Legacy site (assuming the component database has been populated)

- IRMIS has access to the both the .db file that contains the PV fields and the component ports. Spawn an edit process that updates the .db file. This process could be automated, using existing heuristics, to minimize manual effort

- the PV crawler would have to be modified to recognize the component-pin comment field and populate the IRMIS PV database. The correct pv<>pin foreign key relationship would be retained over ioc reboots, where the pv is assigned a new primary key.

Component – PV relationships (proposal)



cmpnt_installation	
cmpnt_installation_id	PK
cmpnt_id	FK
control_p_id	FK
logical_addr	
housing_p_id	FK
locator	
power_p_id	FK
field_name	
channel	

port	
port_id	PK
cmpnt_installation_id	FK
port_type_id	
port_name	
port_order	

pin	
pin_id	PK
port_id	FK
pin_type_id	FK
pin_designator	
pin_usage	
signal_name	
signal_xfmr	

rec	
rec_id	PK
ioc_boot_id	FK
rec_nm	
rec_type_id	
ioc_resource_id	FK

fld	
fld_id	PK
rec_id	FK
fld_type_id	
fld_val	
ioc_resource_id	FK
pin_id	FK

Summary

The IRMIS component/type schema provides an exhaustive description of the accelerator hardware. The schema has been designed to be site neutral – adoption/extension at NSLS2 will be straightforward

A mechanism is required whereby new component-types defined at BNL can be merged into a centralized repository.

A trivial schema change (which will not break existing IRMIS applications) will provide the bidirectional relationships between EPICS PVs and component hardware.