# New KalDet Implementation

**Martin Killenberg**

08. August 2012

# Current Situation

The Micromegas group wants to use TrackMakingKalmanFilterProcessor.

This fails because

- TrackMakingKalmanFilterProcessor uses the lp1jgem KalDetector (hardcoded)

lp1jgem has EVERYTHING hard coded

- 3 modules
- Module position and ID
- Module resolution
- Diffusion constants
- B-Field
- Drift velocity
- Implicit assumptions about content of Gear file
- Overwrites information from Gear file with hardcoded values

# Code Example

## Current code

```
///// FIXME: temporary treatment //////////////////////////////////////
#ifdef APR_2009
    Int_t    module     = superlayer < 1 ? 1 : (superlayer < 2 ? 3 : 6);
#else
#ifdef SEP_2010
    Int_t    module     = superlayer < 1 ? 0 : (superlayer < 2 ? 3 : 5);
#endif
#endif
///////////////////////////////////////////////////////////////////////
```

This is destined to fail!

## Correct code

```
Int_t moduleID = tpcModule->getModuleID();
```

I am currently working on a GearKalDet implementation which instantiates the KalDetector correctly only with information from the Gear file.

# Appeal

- Never hardcode values
  Use Gear, conditions data, config files, Marlin processor parameters

- Do not make code behaviour compiler flag dependent (`#ifdef`)

- Write flexible code
  - You spend 20 % more time developing the code
  - The next guy who want to use it gains 80 % of the time
  - The next to next guy also gains 80 %
  - . . .

  - A few week later you are the next guy!

LCTPC is a collaboration. Let's act as one!