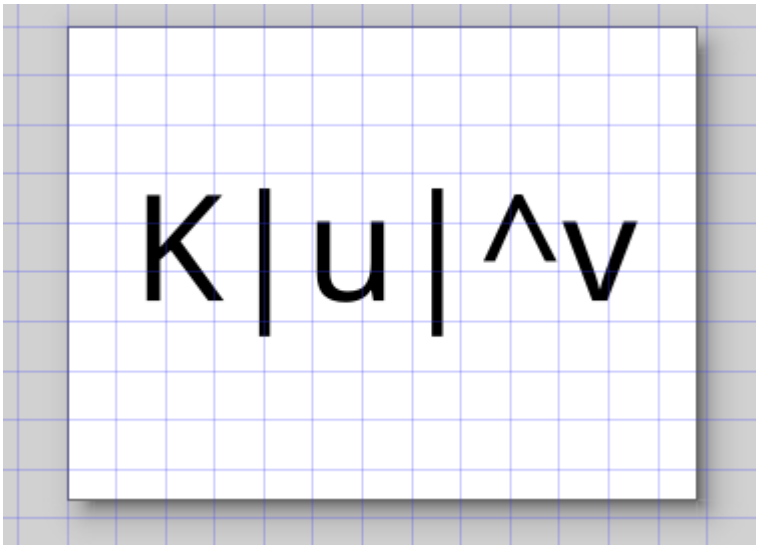
Generate a new block for pysimCoder

Generate the Icon for the new block

Usually I work with Inkscape to generate a new icon, starting from a previous one.

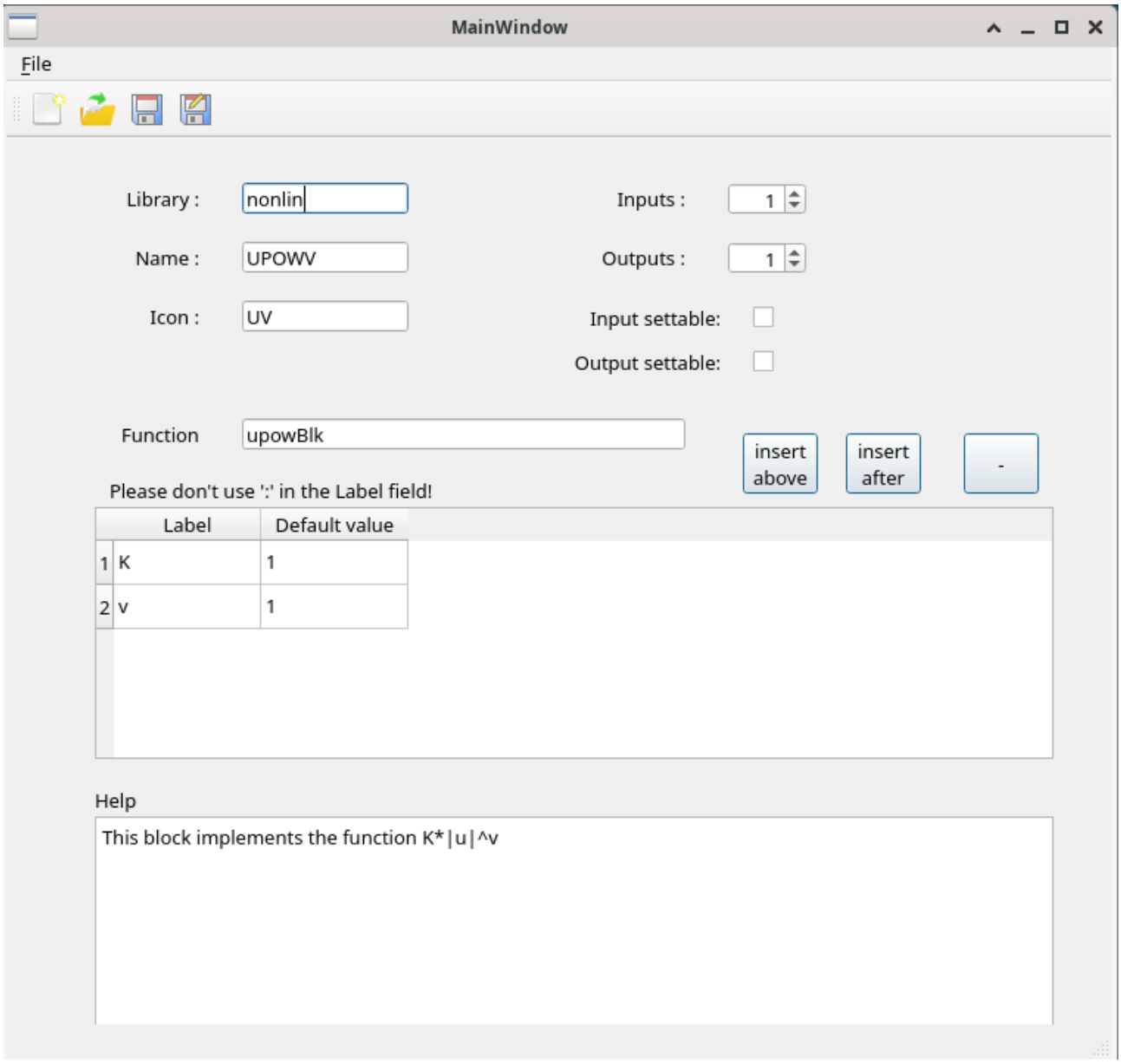


Generate the .xblk file

Launch

$ defBlocks

and fill the fields as shown here



SaveAs -> UPOW.xblk

Generate the python file and the C file

Now launch

$ xblk2Blk

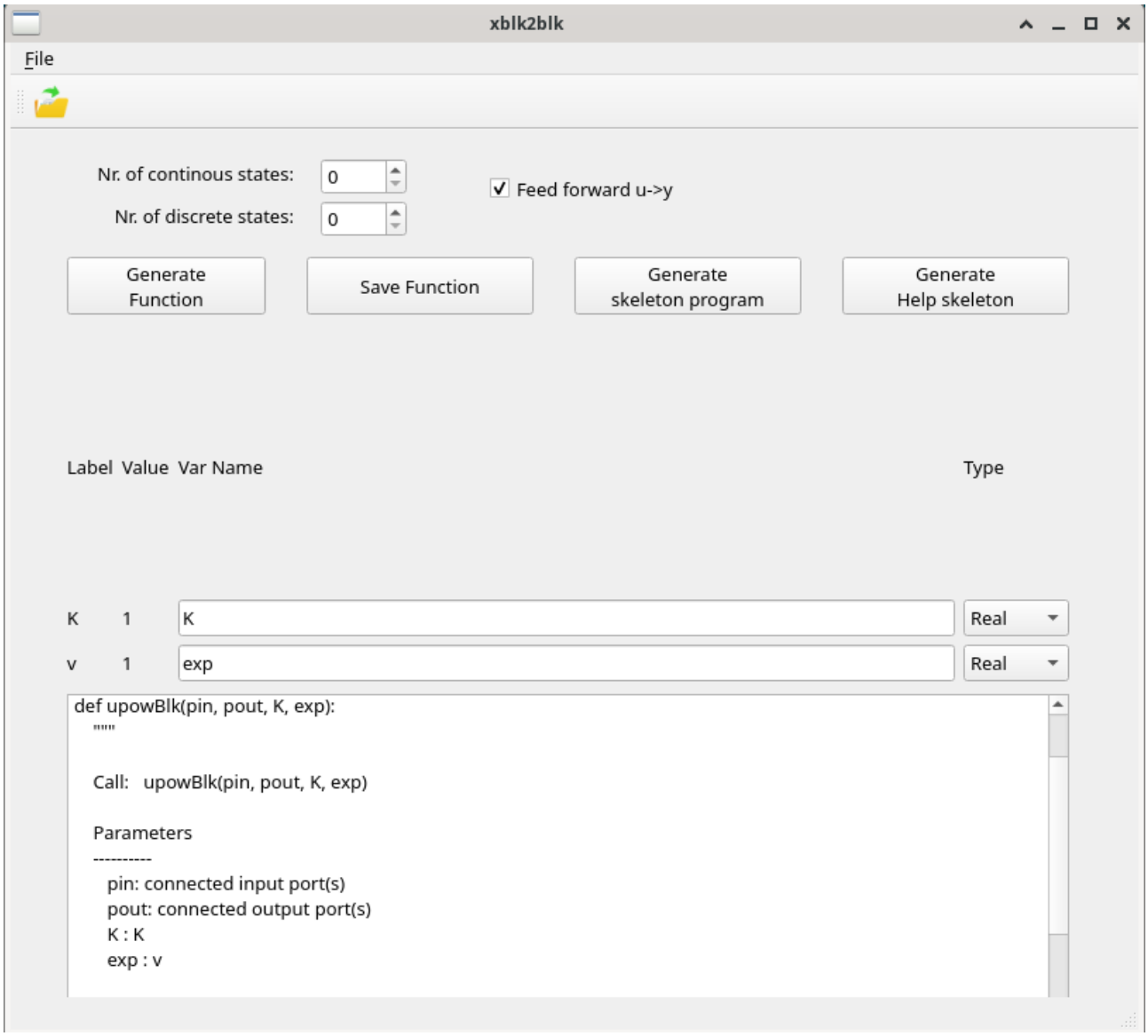
and open the UPOW.xblk file

Then fill the mask with the variable names (K and v) and then click on the buttons

Generate function

Save function

Generate skeleton program



Modify the C file

Open and modify the C code

- Add **#include <math.h>** on the top of the file

```
#include <pyblock.h>
#include <math.h>
```

- Eliminate the "update" function (this block doesn't have internal states!)
- Modify the upow function

```
void upow(int flag, python_block *block)
{
  if (flag==CG_OUT){          /* get input */
    inout(block);
  }
  else if (flag==CG_END){     /* termination */
    end(block);
  }
  else if (flag ==CG_INIT){    /* initialisation */
    init(block);
  }
}
```

- Modify the **init** function (not used for this block)

```
static void init(python_block *block)
{
}
```

- Modify the **end** function (not used for this block)

```
static void end(python_block *block)
{
}
```

- Modify the **inout** function

```
static void inout(python_block *block)
{
  double * realPar = block->realPar;
  double *y = block->y[0];
  double *u = block->u[0];

  double K = realPar[0];
  double v = realPar[1];

  y[0] = K*pow(fabs(u[0]), v);

}
```

Move the generated files and recompile the libraries

Now you have to move the generated files on the right place. This block implements a non linear function, thus we move the files as follow

- UPOW.xblk -> pysimCoder/resources//blocks/blocks/nonlin/
- upowBlk.py -> pysimCoder/resources/blocks/rcpBlk/nonlin/
- UV.svg -> pysimCoder/resources/blocks/Icons/
- upow.c -> pysimCoder/CodeGen/Common/common_dev/

Now you can launch "make" or "make SHV=1 for your targets