Újrakonfigurálható technológiák nagy teljesítményű alkalmazásai

Xilinx System Generator

Szántó Péter

BME MIT, FPGA Laboratórium

Xilinx System Generator

- **MATLAB Simulink Toolbox**
- **Simulink**
  - „Modell alapú” grafikus fejlesztői környezet
  - Esemény-vezérelt szimulátor
  - Párhuzamos végrehajtás modellezése
  - Nagy számú, paraméterezhető blokk
    - Források, nyelők, matematikai függvények
    - DSP blockset, Communication blockset, ….
- **System Generator**
  - Alap FPGA építőelemek (logikai, memória, aritmetika)
  - Komplex építőelemek (CoreGenerator alapokon)

Simulink források

- **Double adattípust szolgáltatnak**
  - Szinusz
  - Ugrásjel, impulzus generátor
  - Háromszög jel
  - Sávkorlátozott fehérzaj
  - Chirp (lineárisan változó frekvenciájú szinusz)
  - Konstans
  - „From file”
  - „From workspace”

Simulink nyelők

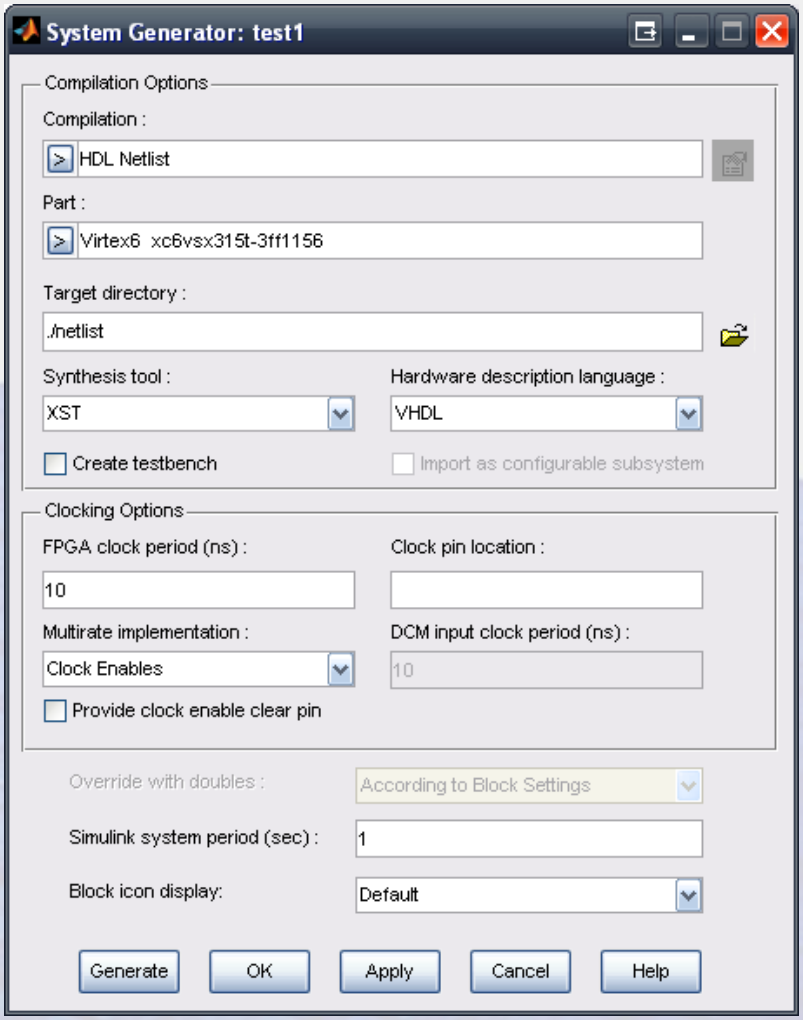
- Display: az aktuális értéket mutatja
- Scope: időbeli hullámforma megjelenítés
- X-Y Graph
- „To file”
- „To workspace”

Mintavételezési periódus

- **Minden Simulink blokk rendelkezik „sample period” paraméterrel**
  - A blokk kimenetének számítási gyakorisága
  - Minden változás a mintavételi pontokban történik
- **System Generator esetén beállítandó**
  - System Generator konfigurációs blokk
  - Bemenetek (Gateway In)
  - Bemenet nélküli blokkok
- **SysGen blokkok esetén a mintavételi periódus meghatározza a létrejövő HW órajelezését**

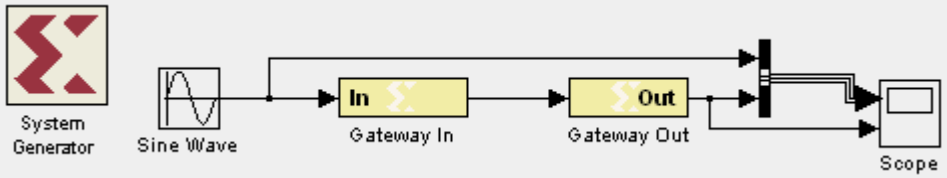
System Generator blokk

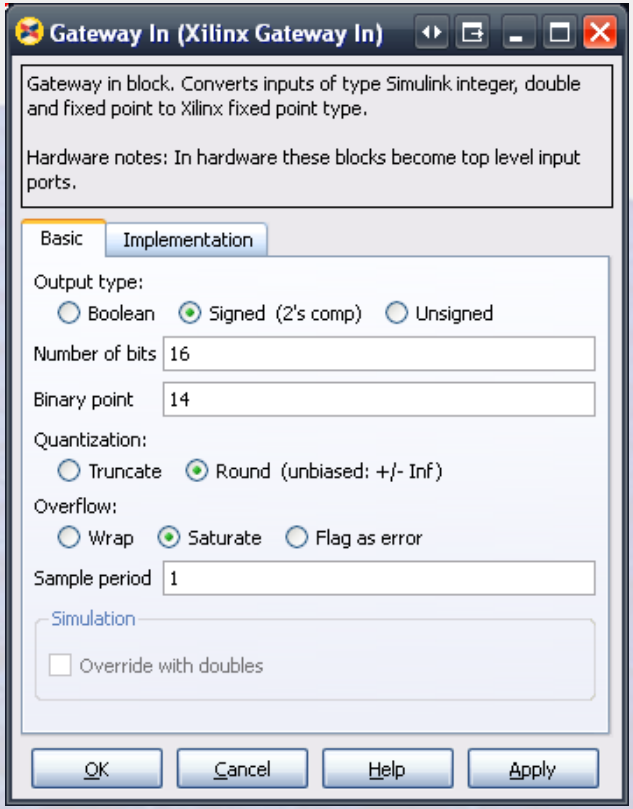
- **Kimenet**
  - HDL, huzalozási lista, bit file
  - HW ko-szimuláció
- **Órajel periódus**
  - Constraint
- **Multi-rate terv**
  - Órajel engedélyezés
  - DCM (nem minden blokkal)
  - Szinkron!
- **Simulink mintavételi periódus**



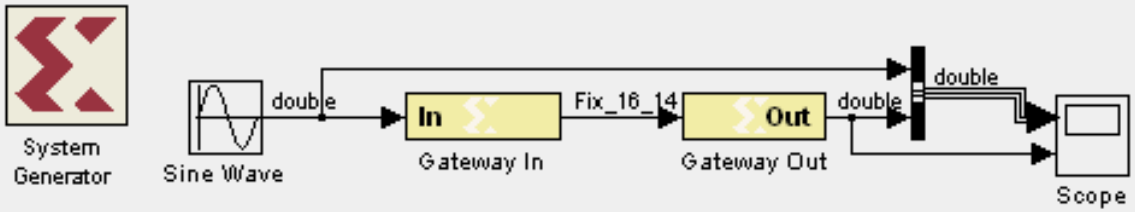
Gateway In/Out

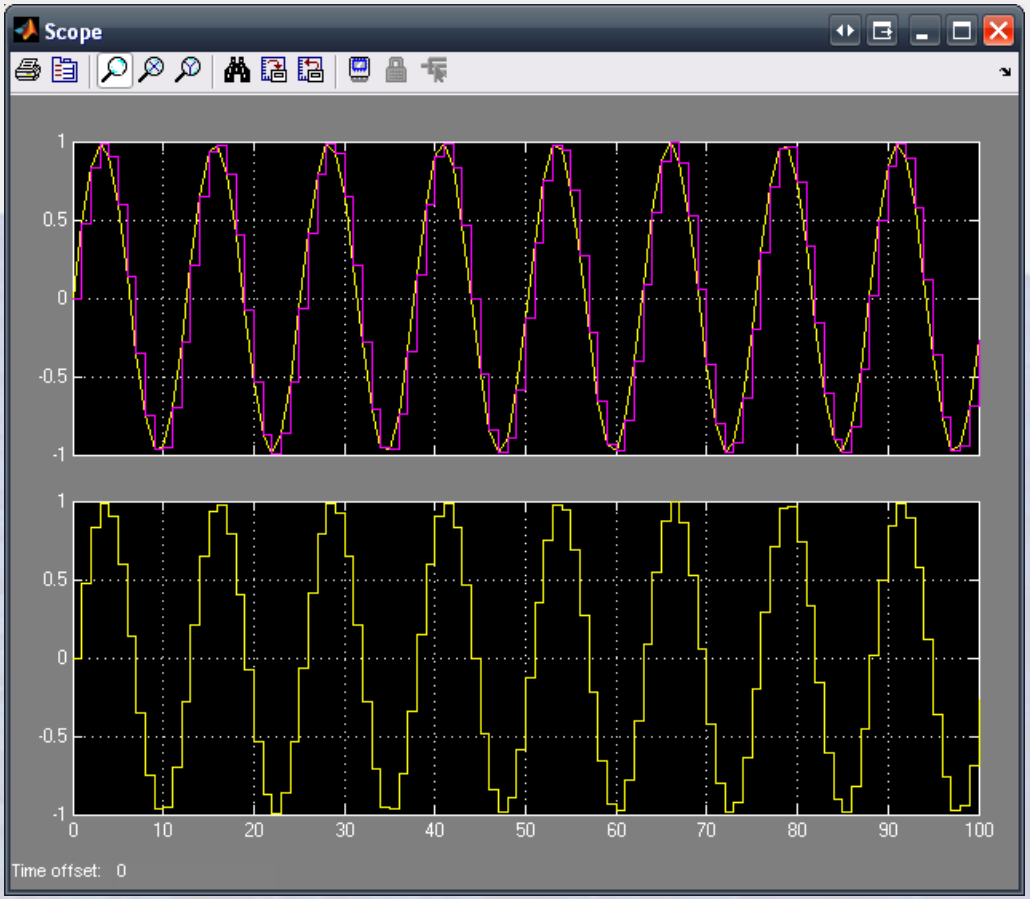
- Az általános Simulink blokkokat a SysGen blokkoktól elválasztó interfész
- Boolean/előjeles/ előjel nélküli
- Matlab double típusainak konverziója
  - Kvantálás
  - Túlcsordulás
- A keletkező HW portjai a Gateway típusának megfelelőek





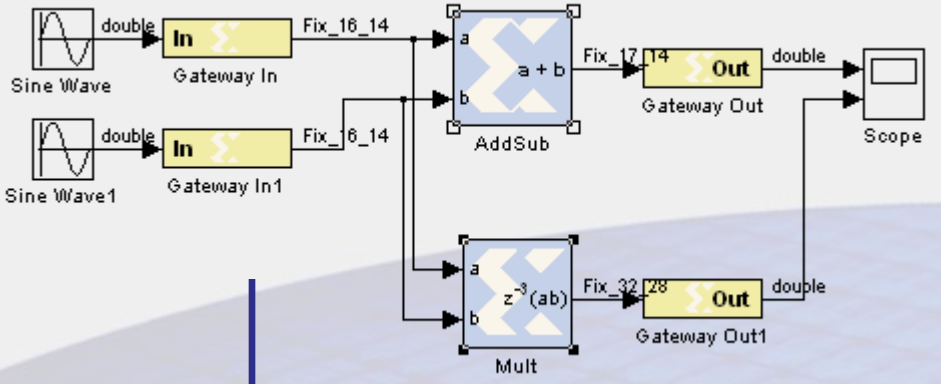
Gateway In/Out

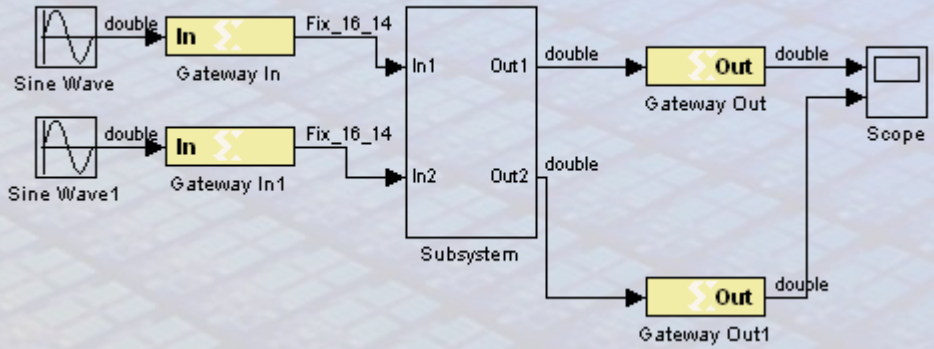


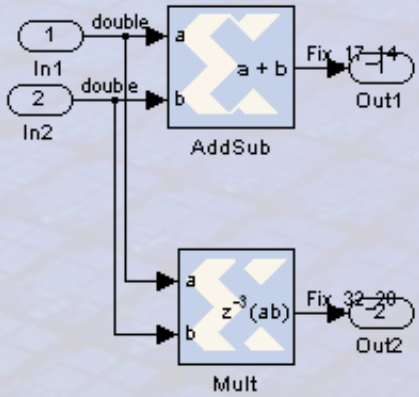


Hierarchia

- Hierarchia elemeinek kijelölése
- Edit/Create Subsystem (Ctrl-G)







Maszkolás

- **Subsystem-ből saját blokkok létrehozása**
  - Paraméterezhető (dialógus ablakkal)
  - Egyedi „Help”
  - Egyedi ikon
  - Védelem
  - Inicializálás
    - Gyakorlatilag Matlab kóddal a szükséges modell változók beállítása
    - Az inicializálás után az itt használt változók megszűnnek

Számábrázolás

- **Boolean**
  - Tipikusan vezérlő bemenetekhez (pl. CE, RST)
  - 0 vagy 1 (nincs undefined állapota)
- **FIX & UFIX: tetszőleges szélességű fix pontos**
  - Blokk kimenete
    - „FULL” → a bemeneti operandusok alapján
    - „User defined” → tetszőlegesen állítható
      - Kvantálás: csonkolás vagy kerekítés
      - Túlcsordulás: átfordulás vagy szaturáció
      - VAN hardver vonzata!!

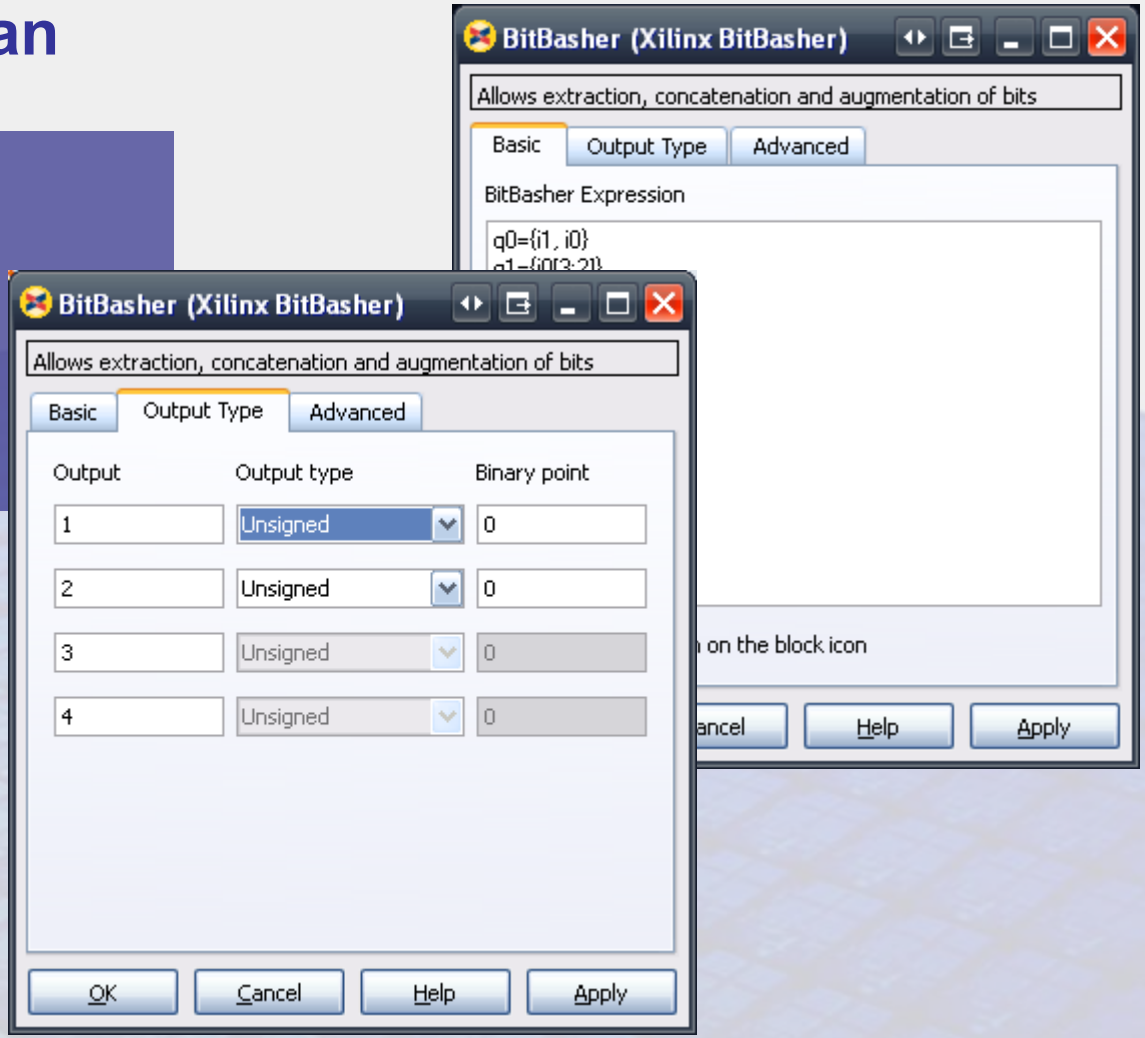
Konvertáló blokkok

- **Reinterpret**
  - Az aktuális bitminta megváltoztatása nélkül típust vált (a bitszélesség NEM változik)
  - „Logikai konverzió” → nincs hardver vonzata
- **Convert**
  - Típuskonverzió (kvantálás + túlcsordulás kezelés)
- **Concat**
- **Slice**
- **BitBasher**
  - Konkatenálás, bit kiválasztás ~Verilog szintaxissal
  - Javasolt az előbbi kettő helyett

BitBasher blokk

- Tetszőleges számú be- és kimenet
- Új kimenet új sorban

```
q0 = {i2, i1, i0}
q1 = {i0[7:3]}
q3 = {i0[0:7]}
q4 = {4{i1, i0}}
```



Expression blokk

- **Bitműveletek elvégzése**
- **Verilog operátorok: &, |, ~, ^**
- **Max. 16 bemenet**
  - Bemenetek száma a beírt kifejezés alapján automatikus
  - „Align binary point”: bemenetek bináris pontjainak automatikus illesztése
    - Ha ki van kapcsolva, minden bemeneti változóban azonos helyen kell legyen

Mcode blokk

- **Matlab kód részleges támogatása HW szintézishez**
  - NEM algoritmikus leírásra való
- **A blokk ki és bemenetei Xilinx fix pontos típusúak**
  - Legalább egy be-, és egy kimenet kell
- **Támogatott operátorok**
  - Egyszerű if-else szerkezetek
  - Összeadás, kivonás, szorzás
  - Shiftelés (xl_lsh(), xl_rsh()), komparálás
  - Logikai operátorok (&, |, ~),
  - Bitműveletek (xl_and(), xl_or(), …)
  - xl_concat(), xl_slice()
  - Állapotváltozó: xl_state()

Mcode – xl_state

- „persistent” változó: megtartja az értékét a szimulációs lépésközök között
- Kezdőértékét az xl_state() függvénnyel kapja

```
persistent s, s = xl_state(0, {xlUnsigned,2,0});
```

- Vektor változó deklaráció (max méret: 4)

```
persistent s, s = xl_state(zeros(1,4),
                           {xlUnsigned,2,0}, 4);
```

- Vektor metódusok
  - Címzés: var(addr)
  - var.push_front(in): in betöltése, vektor hosszának növelése
  - var.push_front_pop_back(in)

Mcode - állapotgép

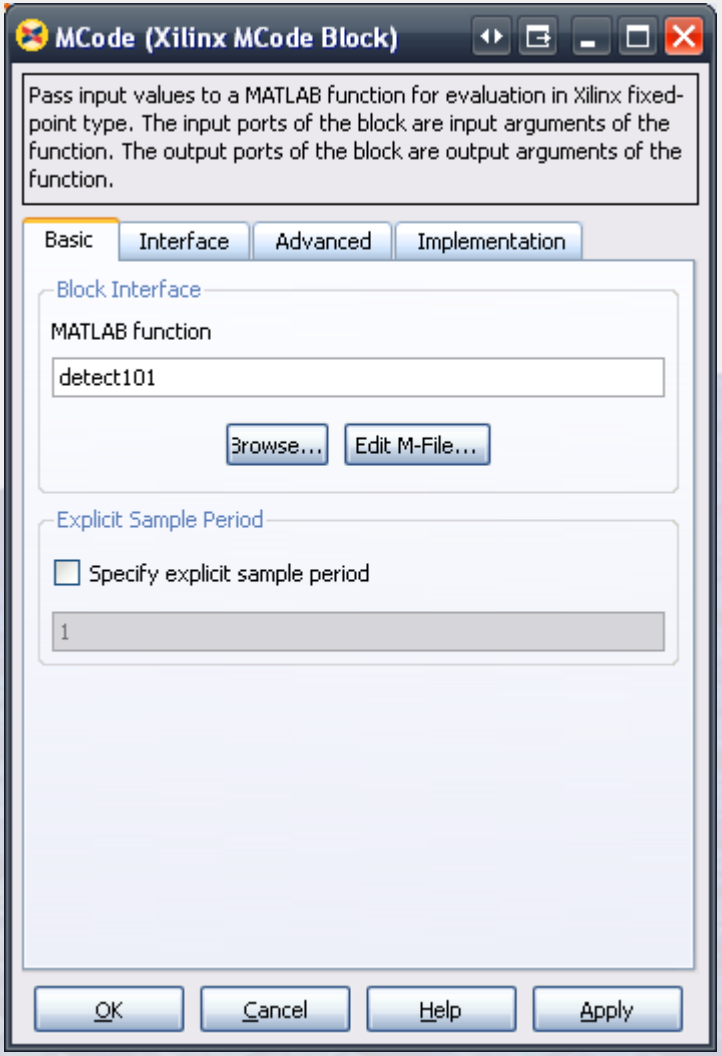
- Állapotváltozó: persistent; kezdeti állapot: xl_state()
- Pl. „101” bitminta detektálás

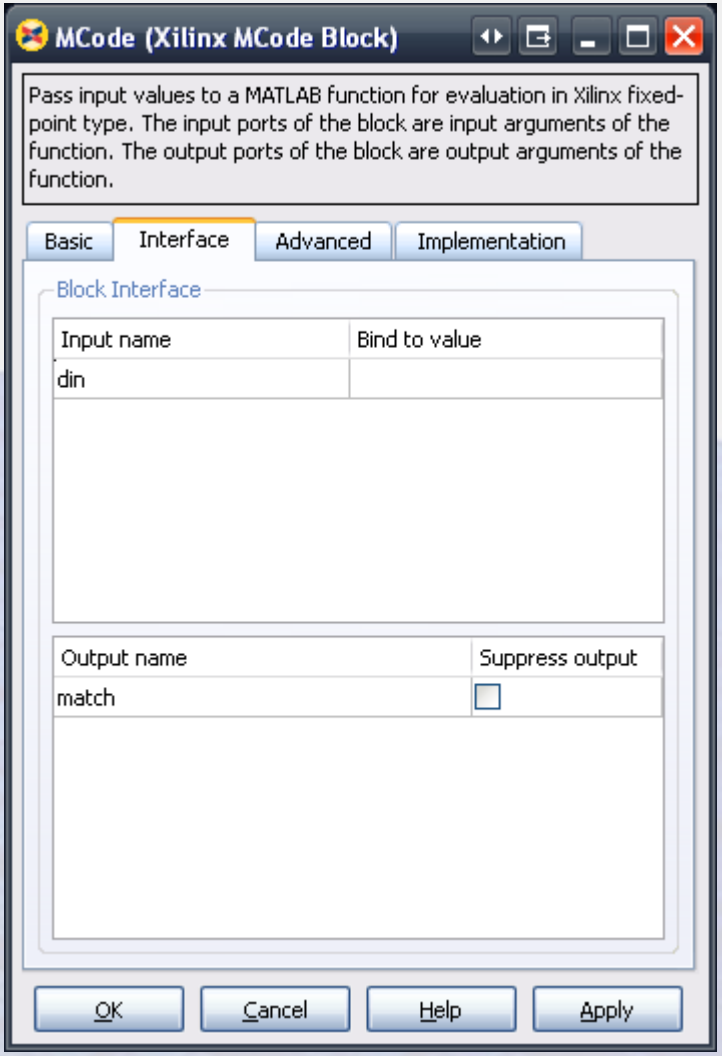
```
function match = detect101(din)
  persistent s, s=xl_state(0, xlUnsigned,2,0));
  match = 0;

switch s
  case 0
    if din==1 s = 1;
    else      s = 0; end
  case 1
    if din==0 s = 2;
    else      s = 1; end
  case 2
    if din==1 s = 1;  match = 1;
    else      s = 0;  match = 0; end
end
```

Állapotváltozó

Állapot átmenet

Mcode – beállítások



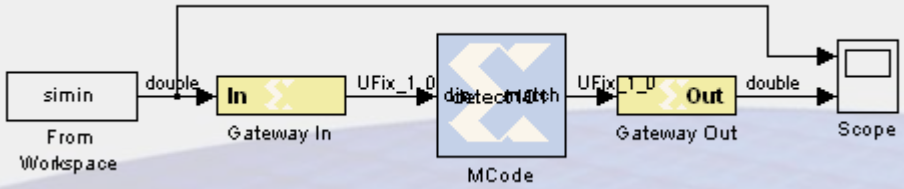


Mcode – példa

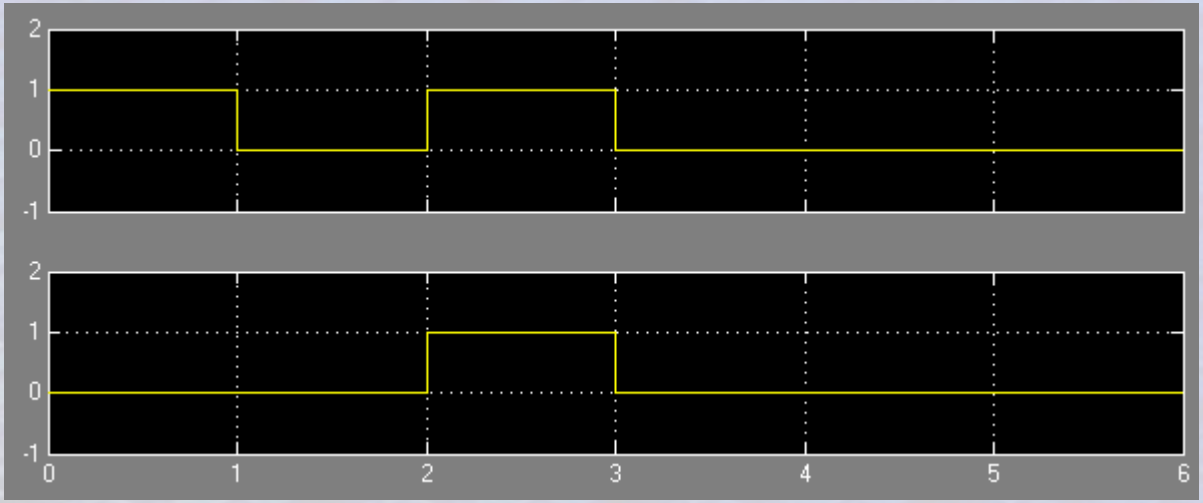
- **Az előző állapotgép bemenő jele**

```
t = 0:1:5; v = zeros(1,6); v(1)=1; v(3)=1;
simin = [t', v'];
```

- **Modell**



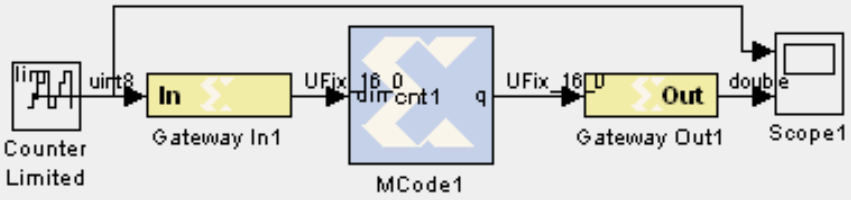
- **Szimuláció**

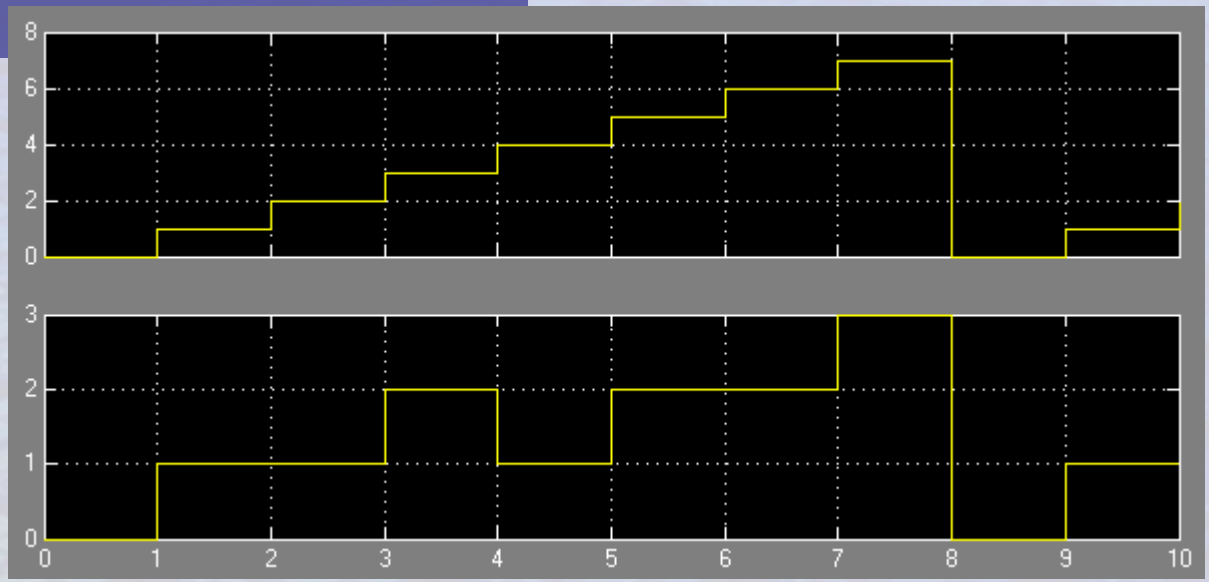


Mcode – aritmetika

- ‘1’ bitek számlálása

```
function q = cnt1(din)
  q = xl_slice(din, 0, 0);
  for i=1:15
      q = q +
        xl_slice(din, i, i);
  end
```





Mcode – példák

- Shift regiszter

```
function q = delay(d, lat)
  persistent r, r = xl_state(zeros(1, lat), d, lat);
  q = r. back;
  r.push_front_pop_back(d);
```

- Címezhető shift regiszter (→ SRL)

```
function q = addrsr(d, addr, en, depth)
  persistent r, r = xl_state(zeros(1, depth), d);
  q = r(addr);
  if en
    r.push_front_pop_back(d);
  end
```

Mcode – példák

- ROM

```
function q = addrsr(contents, addr, arith, nbits, binpt)
  proto = {arith, nbits, binpt};
  persistent mem, mem = xl_state(contents, proto);
  q = mem(addr);
```

- RAM

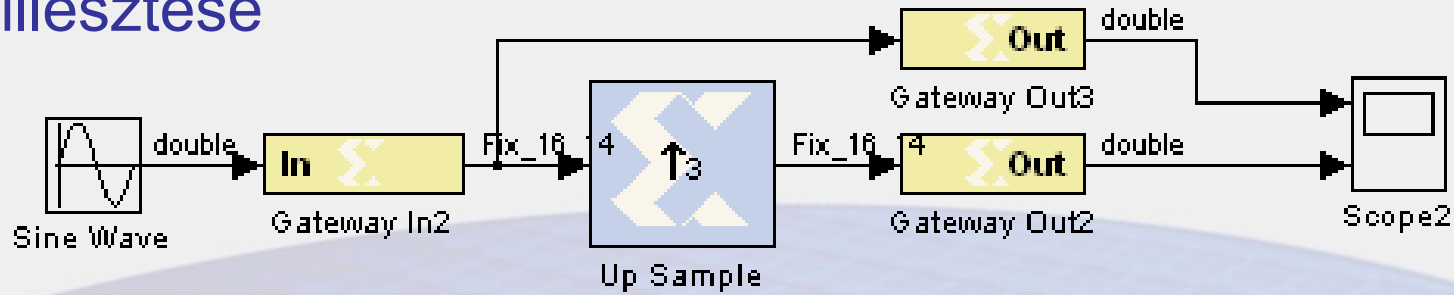
```
function dout = ram(addr, we, din, depth, nbits, binpt)
  proto = {xlSigned, nbits, binpt};
  persistent mem, mem = xl_state(zeros(1, depth), proto);
  q = r(addr);
  if we
    r(addr) = din;
  end
```

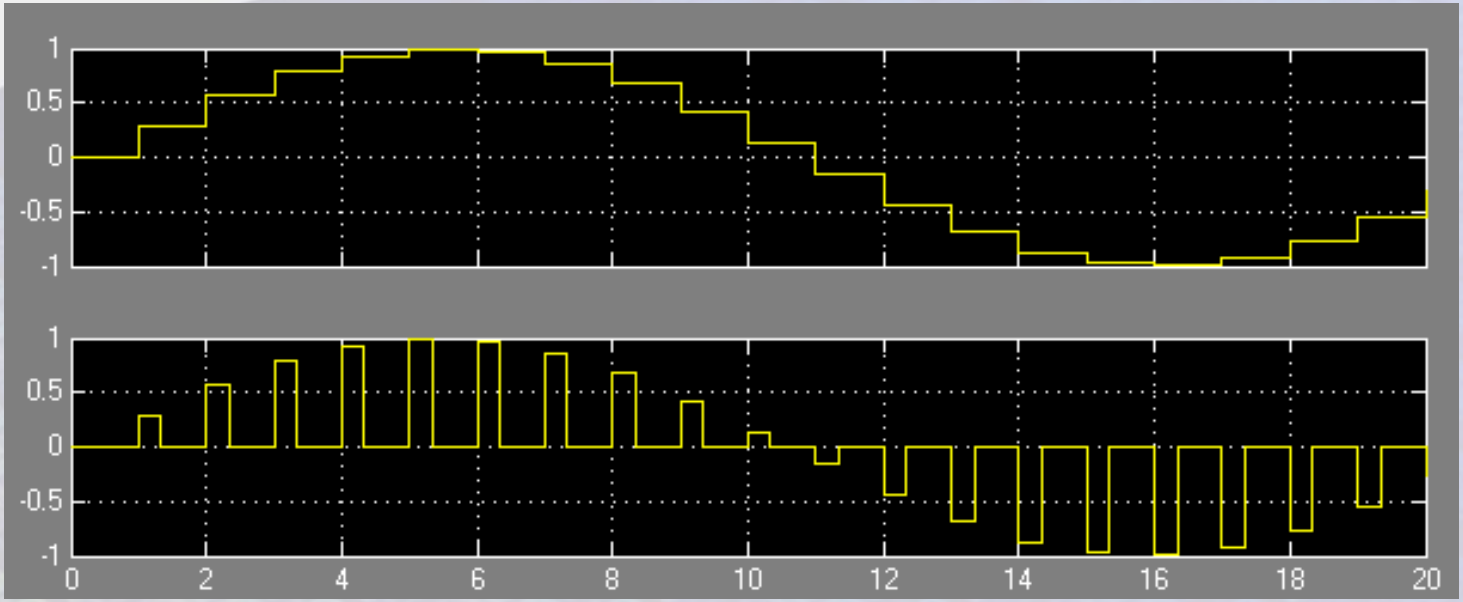
Multi-rate rendszerek

- **SysGen-ben egyetlen, közös időalap van**
  - Sample Rate
- **Mindennek a működése ezen alapul**
  - Órajel engedélyezéssel
  - Egymással szinkron órajelekkel
- **Néhány blokk mintavételi frekvencia változást okoz**
  - Up Sample, Down Sample
  - Time division multiplexer
  - Serial to Parallel, Parallel to Serial
- **A rendszerben az összes mintavételi periódus egész számú többszöröse a rendszer mintavételnek (SysGen doboz)**

Up Sample

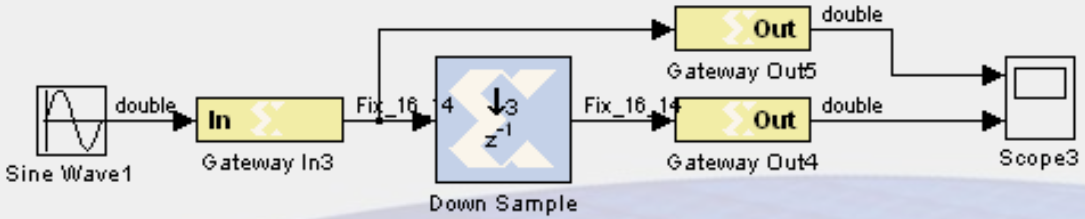
- **Up Sample: mintavételi frekvencia növelés**
  - Előző minta tartása
  - 0 beillesztése

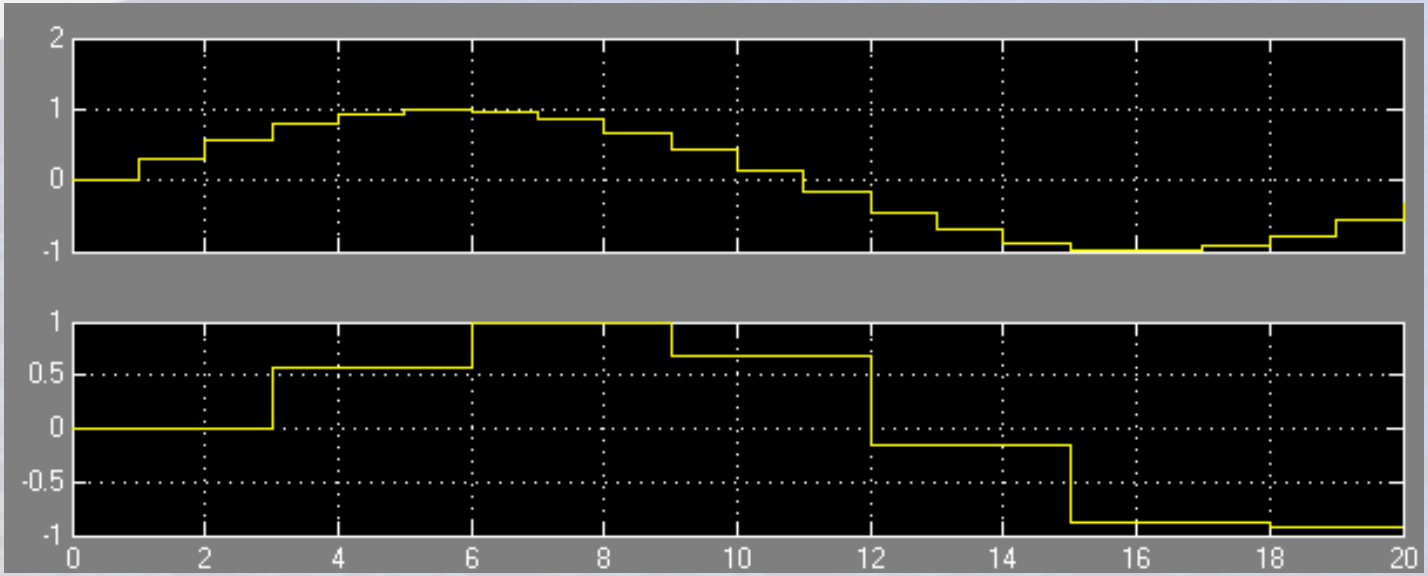




Down Sample

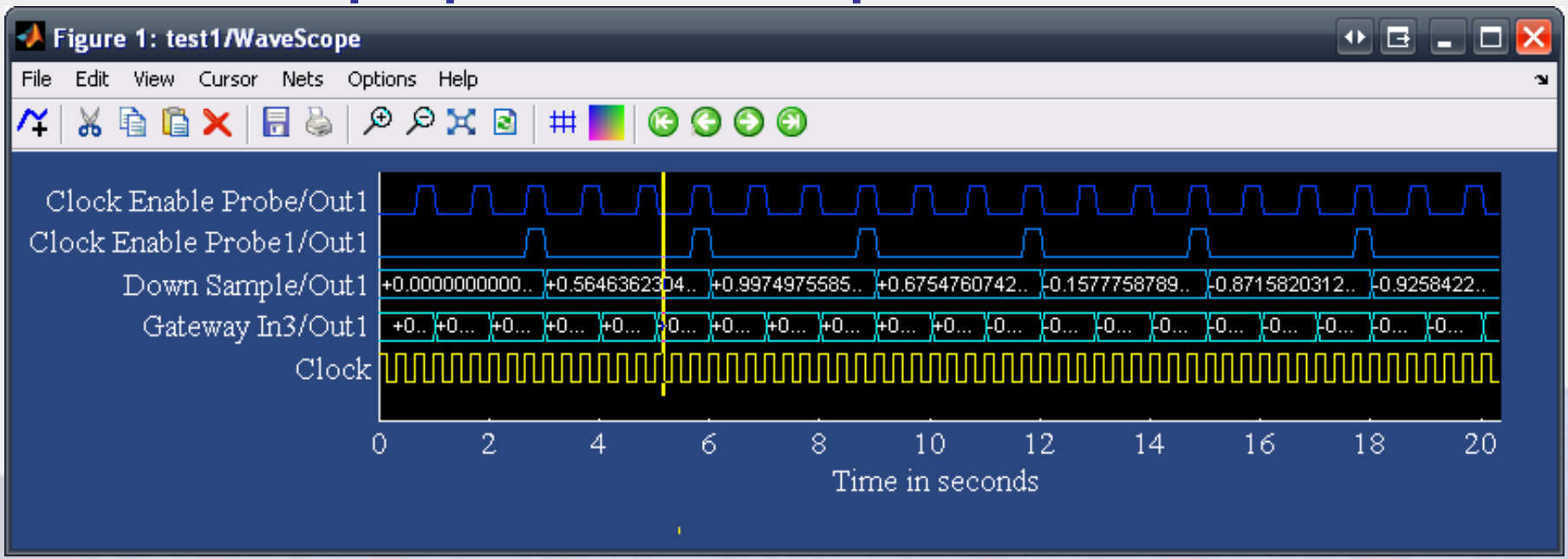
- **Mintavételi frekvencia csökkentése**
  - Első vagy utolsó minta tartásával





Clock enable

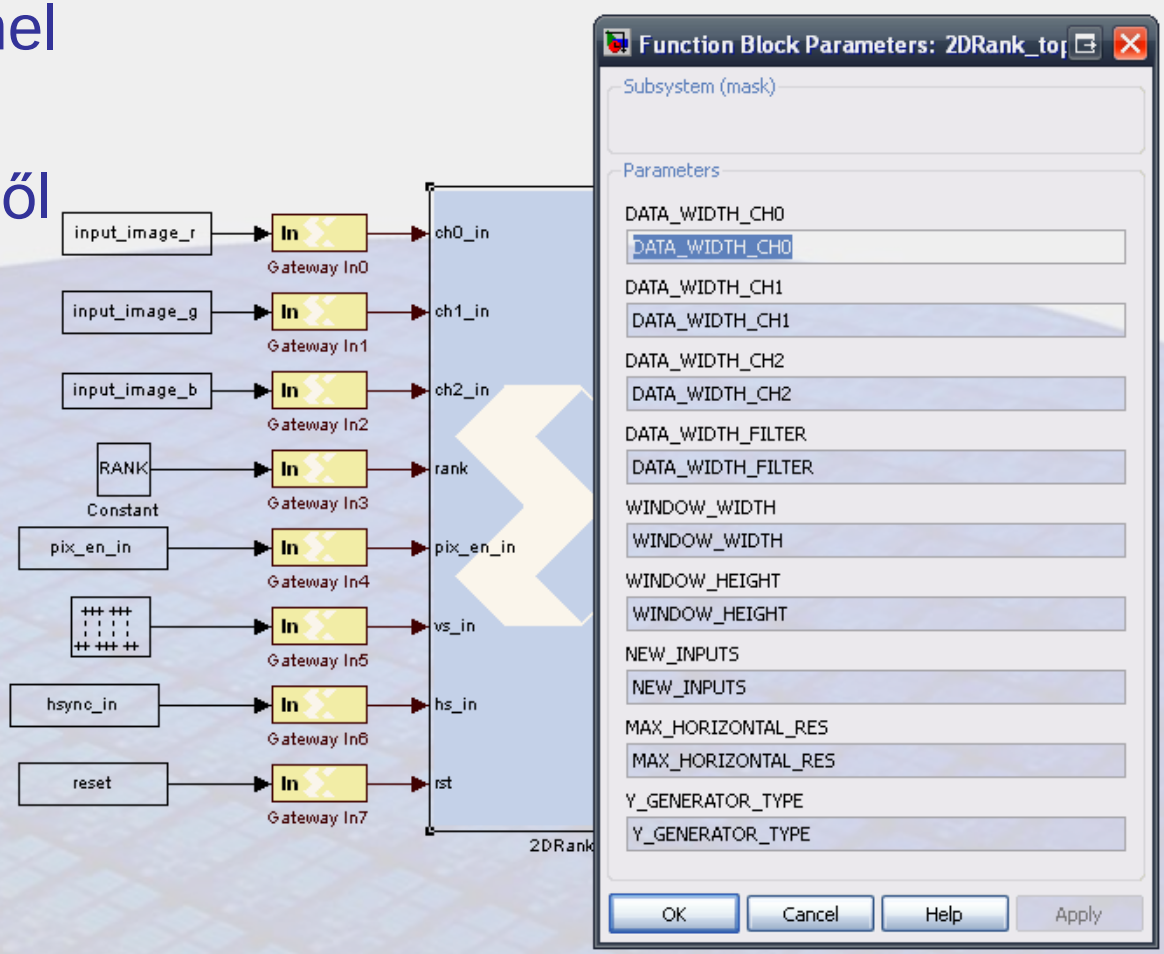
- Down Sample példa WaveScope-ban



- Ritkább CE-vel engedélyezett FF-ok: „multi-cycle path” constraint → SysGen generál constraint file-t

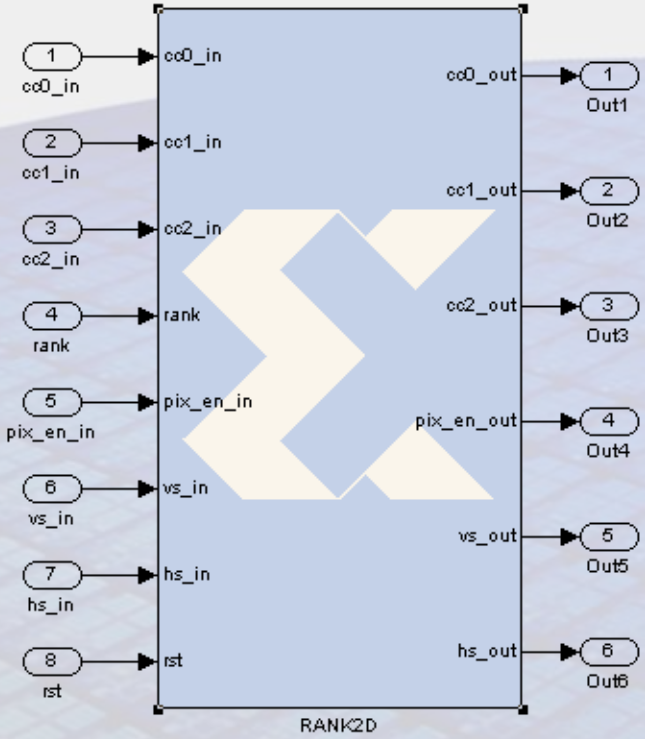
Black Box (1)

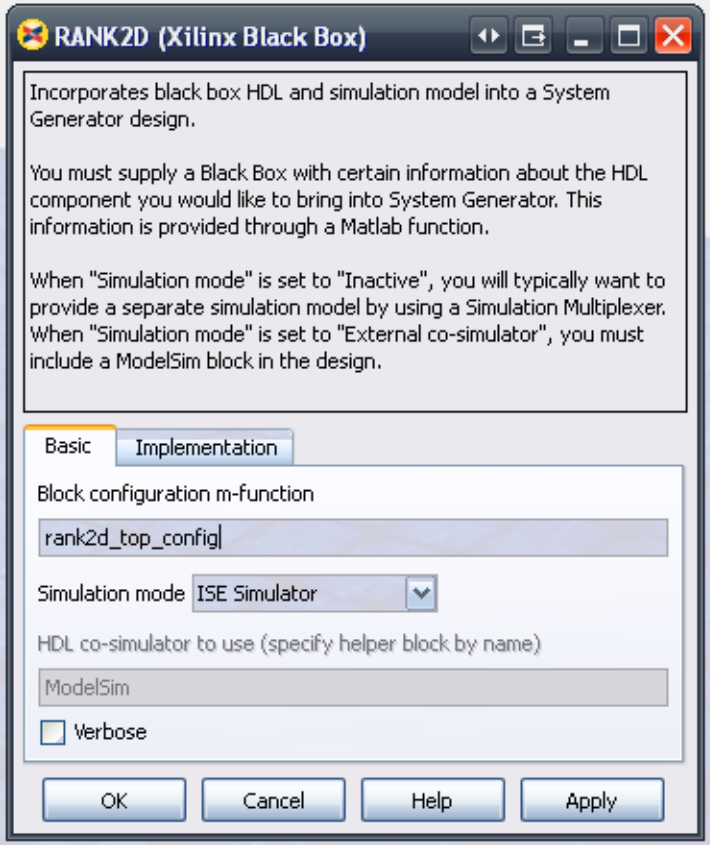
- **Tetszőleges HDL kód beilleszthető „Black Box”-ként**
  - Szimuláció Isim-mel vagy ModelSim-mel
  - Paraméterezhető HDL kódok GUI-ből állíthatók



Black Box (2)

- **Black Box konfigurációja M file-ból**
  - Paraméterek ellenőrzése
  - Ki- és bemeneti portok megadása
  - HDL file-ok megadása





Szimuláció

- **Szoftveres Simulink**
  - Gyorsabb mint HDL (kivéve: Black Box)
- **Hardware koszimuláció**
  - JTAG vagy Ethernet kapcsolat
  - PC – FPGA adatmozgatás lassú lehet
  - Gyorsítás
    - Shared Memory: FPGA belső memória Matlab adatterületre leképezve
    - Buffer-elt bemeneti és kimeneti adattömbök mozgatása → burst átvitel