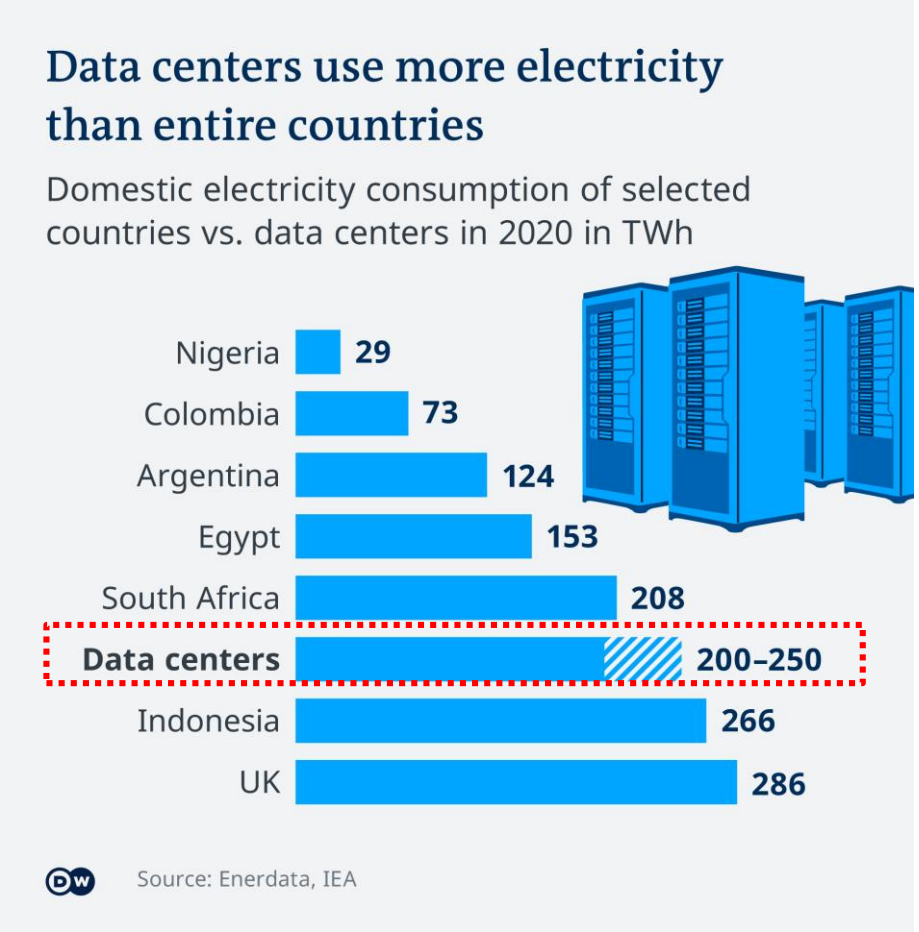
Low Sensitivity Optical Receiver and Berkeley Analog Generator (BAG)

Summer Seminar

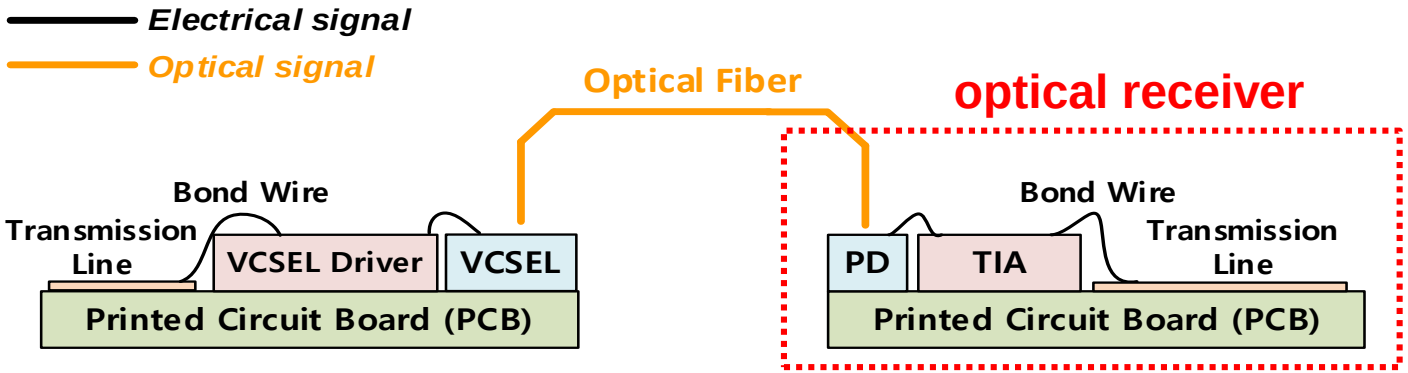
TaeYoung Choi

Low Sensitivity Optical Receiver

- Why Low Sensitivity ?

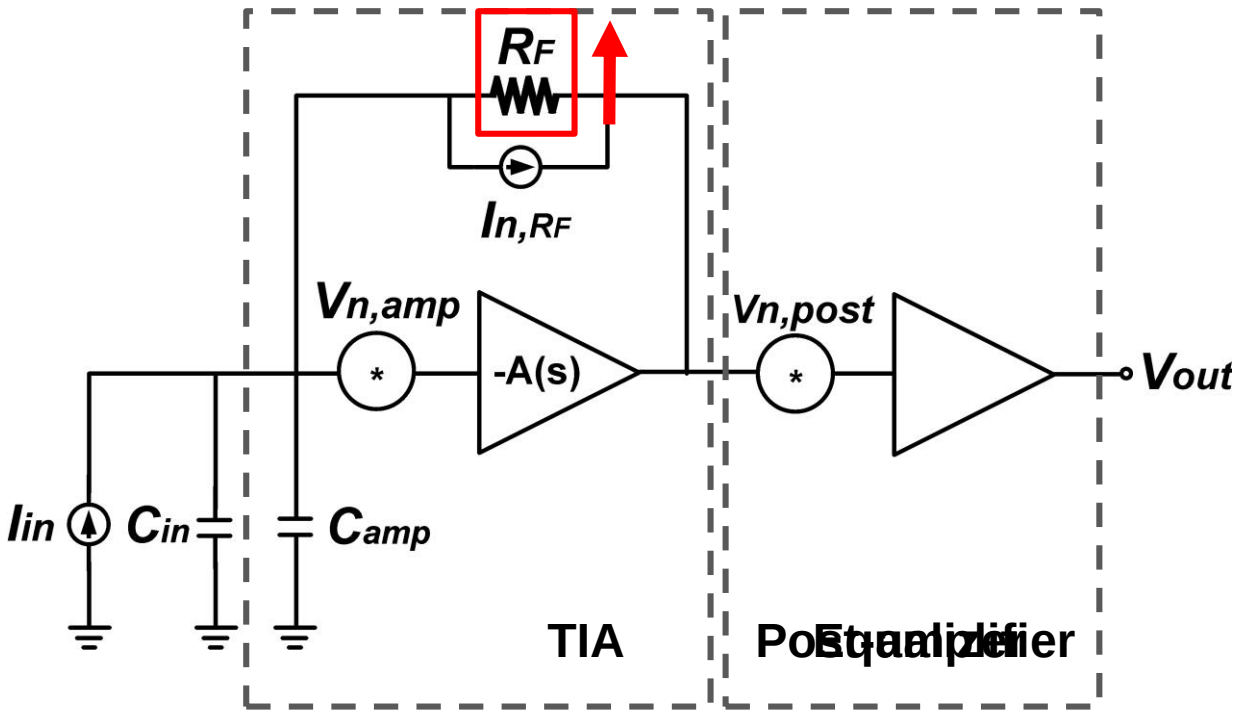


Optical Sensitivity : minimum p2p optical power to achieve a specified BER



< VCSEL based optical interconnect >

Low Sensitivity Optical Receiver



< optical receiver front-end noise model >

< Optical Sensitivity >

$$P_{sens} = \frac{2Qi_n^{rms}}{R}, where\ R = responsivity\ of\ PD$$

< Input-referred noise PSD >

$$I_{n,TIA}^2(f) = \underbrace{\frac{4kT}{R_F}}_{I_{n,RF}^2} + \underbrace{\left[\frac{1}{R_F^2} + \omega^2\left(C_{in} + C_{amp}\right)^2\right] v_{n,amp}^2}_{I_{n,amp}^2} + \underbrace{\frac{v_{n,post}^2}{|Z_T(s)|^2}}_{I_{n,post}^2}$$

$$Z_T(s) = -R_T \frac{1}{1 + s/w_p} \qquad R_T = \frac{A}{A+1} R_F$$

$$BW_{3dB} = \frac{\omega_p}{2\pi} = \frac{A+1}{2\pi R_F C_{total}}$$

Input-referred noise PSD ↓

Transimpedance Gain ↑ -3dB Bandwidth ↓

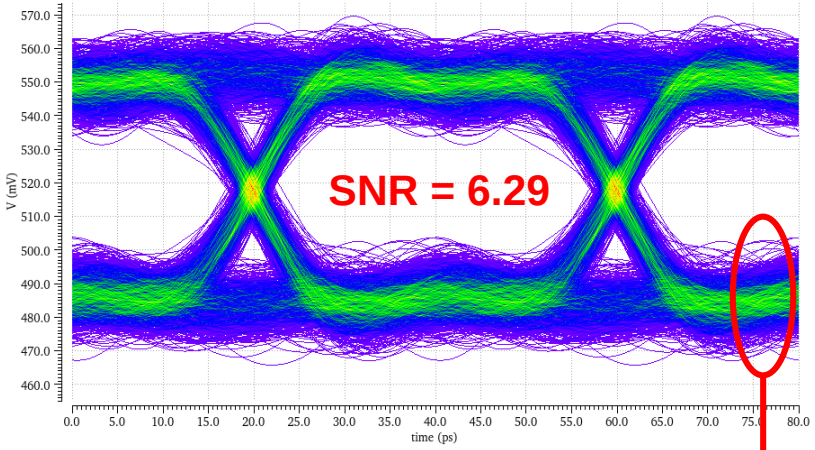
Low Sensitivity Optical Receiver

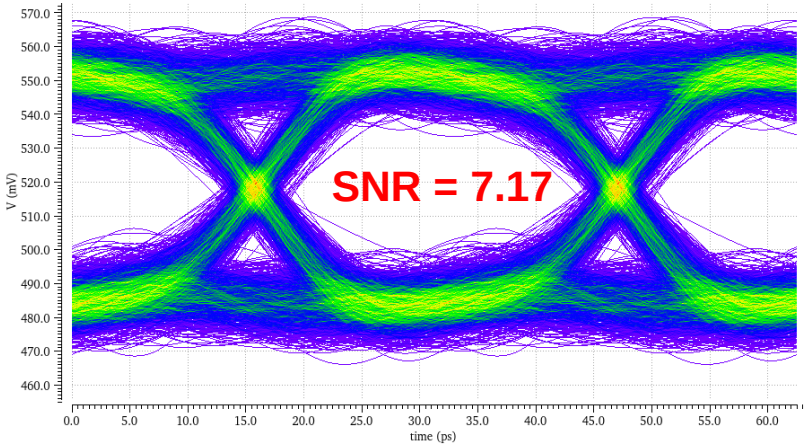
25Gbps

32Gbps

< Original >

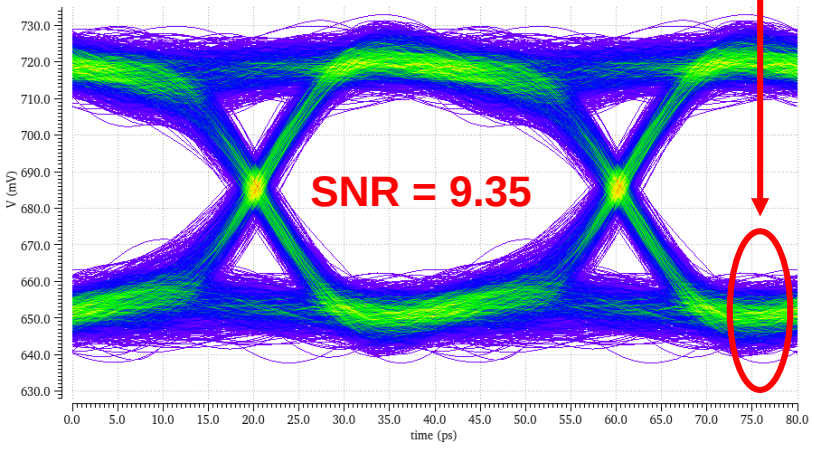
4.963mW

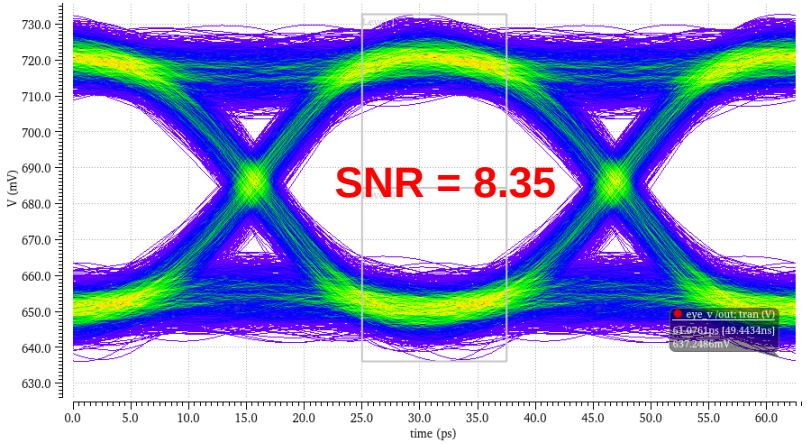




< TIA + EQ >

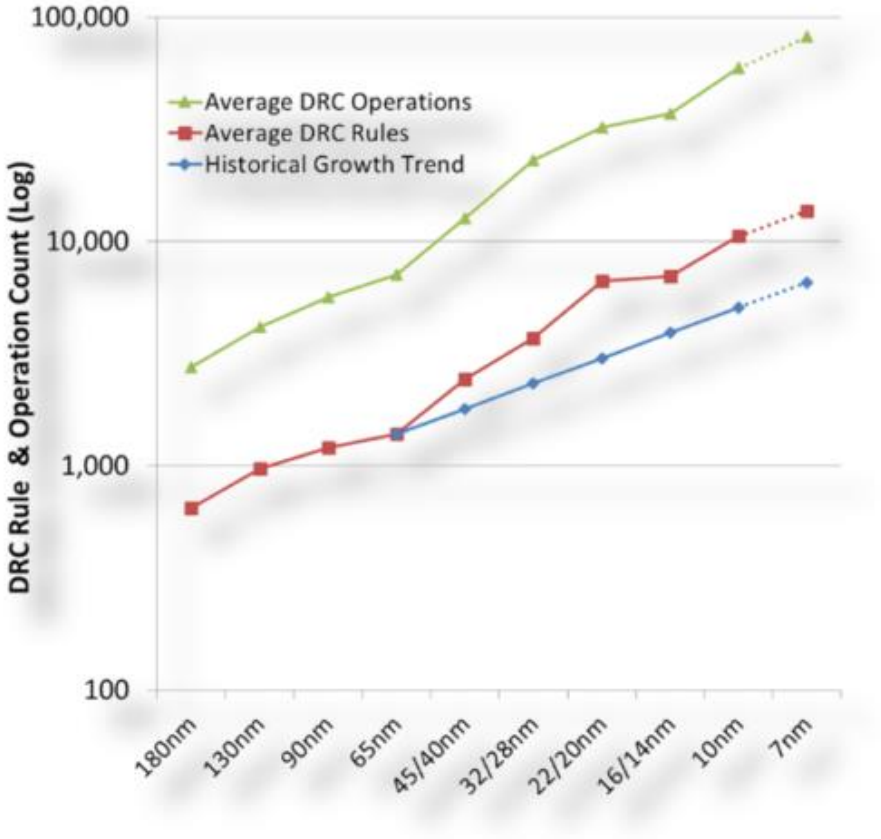
3.061mW



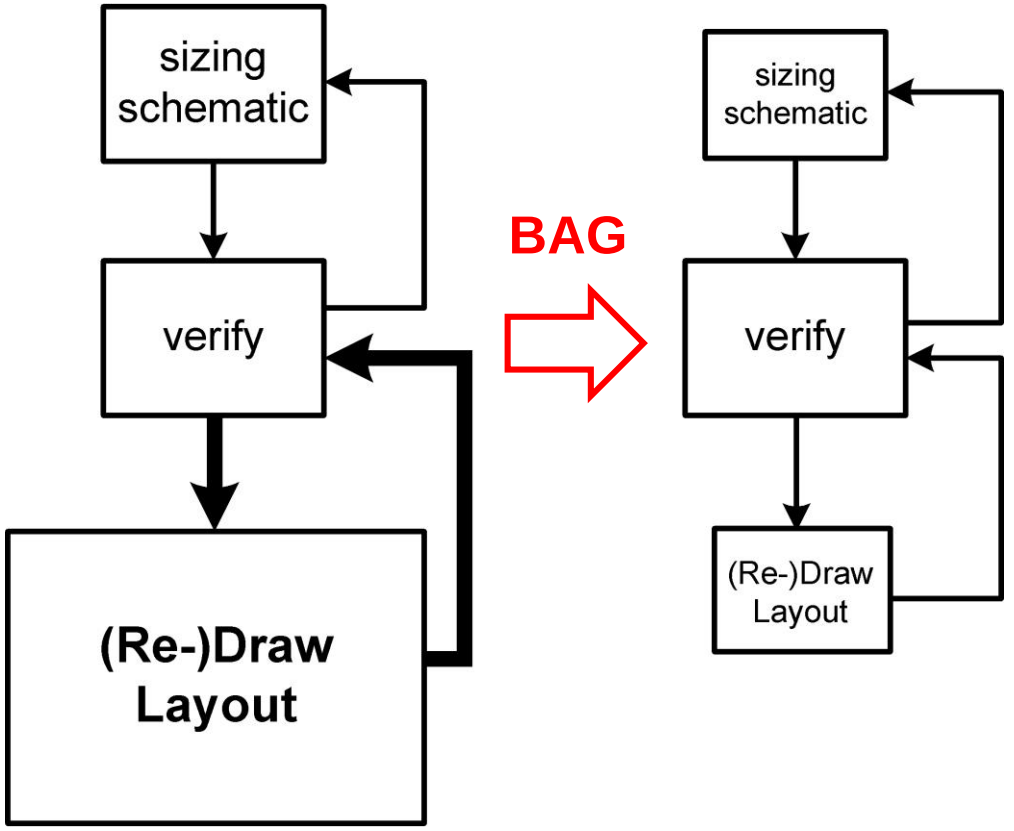


Berkely Analog Generator

- Why BAG ?



source : SIEMENS



Berkely Analog Generator

```python
if not unit_mode:
    # get dimensions based on resolution
    radius = int(round(radius / res))
    width = int(round(width / res)) // 2 * 2  # make width to be even number
    spacing = int(round(spacing / res))
    opening = int(round(opening / res)) // 2 * 2
    via_width = int(round(via_width / res)) // 2 * 2

# inductor radius of this turn
step_phase = 2 * math.pi / n_side
init_phase = -math.pi / 2 + step_phase / 2
turn_rad = radius - turn * (width + spacing) / math.sin(-init_phase)

if turn == 0 and n_turn == 1:  # outer path with only 1 turn inductor (mode 1
    path_coord, lead_coord, tail_coord, top_coord, bot_coord, via_coord = \
        self.get_octpath_coord(res, turn_rad, n_side, width, spacing,
                               opening, 0,  via_width, mode=1, unit_mode=True)

elif turn == 0 and n_turn > 1:  # outer path with more than 1 turn inductor (
    path_coord, lead_coord, tail_coord, top_coord, bot_coord, via_coord = \
        self.get_octpath_coord(res, turn_rad, n_side, width, spacing,
                               opening, 0, via_width, mode=2, unit_mode=True)
```

< layout generator code >

```yaml
params:
    n_turn: 4
    layid: 10
    radius: 52  # 31.5
    spacing: 4 # 1 # 2
    width: 3.6 # 3.6
    opening: 20 # 20, have a try with 12
    via_width_ratio: 1
    tap_len: 10
    lead_len: 10
```

< size parameters file >