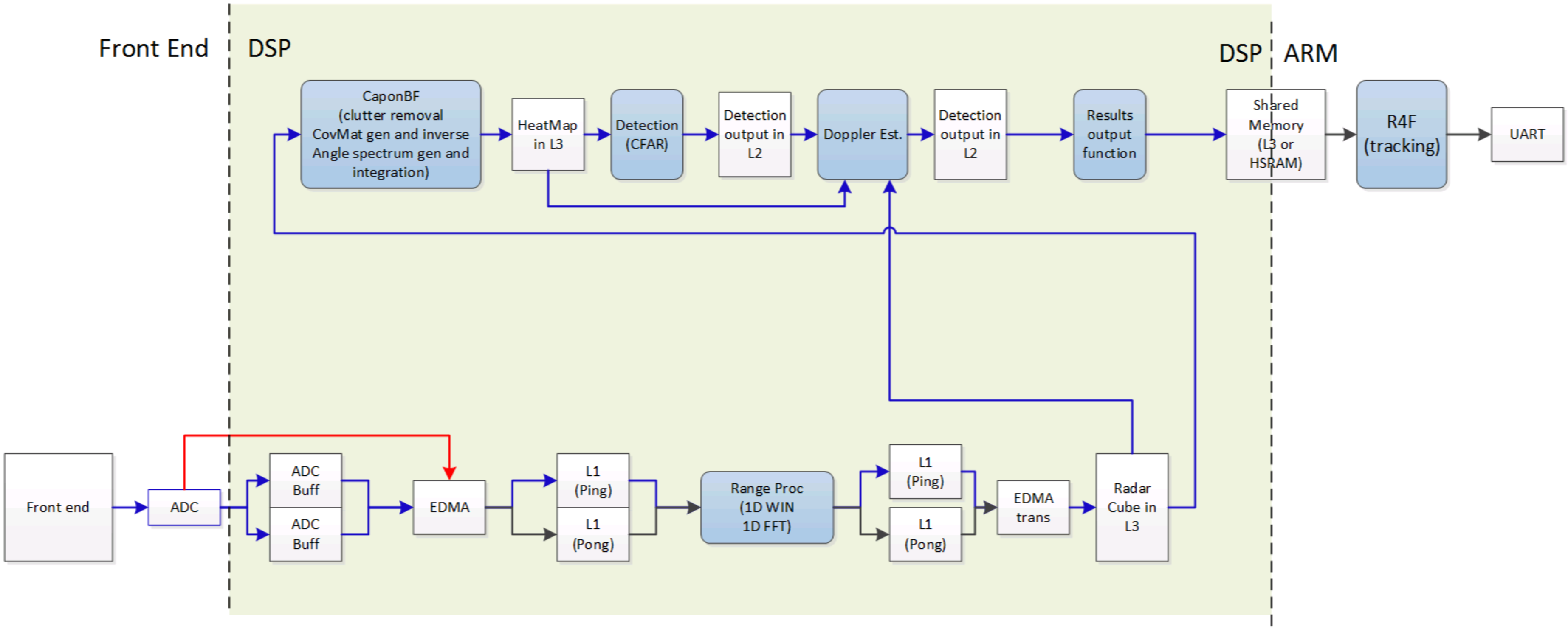
People Counting Demo: Algorithm and Implementation for Low Level Signal Processing Chain

Low Level Signal Processing Chain: Block Diagram



- Low level signal processing chain is implemented on DSP (C674x) of IWR68xx
- Low level signal processing chain outputs detected point cloud with information of range, Doppler, angle, and detection SNR.
- Output point cloud information are shared with R4/tracker in L3 memory for the current implementation.

Low Level Signal Processing Chain: Front End and Range Processing

- Front-end
  - ADC samples are EDMAed into local L1 scratch buffers in PING-PONG fashion.
- Range processing:
  - 1D windowing function
  - 16-bit fixed point FFT per antenna per chirp.
  - Results are first stored in local scratch buffers then EDMAed to L3 radar cube with transpose.
  - In L3, the data is order in the following way:

S[rangeBin0, ant0, chirp0], S[rangeBin0, ant0, chirp1], S[rangeBin0, ant0, chirp2], …, S[rangeBin0, ant0, chirp(Nc-1)],
S[rangeBin0, ant1, chirp0], S[rangeBin0, ant1, chirp1], S[rangeBin0, ant1, chirp2], …, S[rangeBin0, ant1, chirp(Nc-1)],
…
S[rangeBin0, ant7, chirp0], S[rangeBin0, ant7, chirp1], S[rangeBin0, ant7, chirp2], …, S[rangeBin0, ant7, chirp(Nc-1)],
…
S[rangeBin(N-1), ant0, chirp0], S[rangeBin(N-1), ant7, chirp1], S[rangeBin(N-1), ant0, chirp2], …, S[rangeBin0, ant7, chirp(Nc-1)]

Low Level Signal Processing Chain: Capon Beamforming

- Clutter removal
  - Per range bin per antenna, remove static clutter using DC removal scheme
  - This removes the static objects
  - Also removes objects moving completely tangentially with regard to the sensor which show up in 0 Doppler bin, similar to true static objects.
- Covariance matrix generation and inverse:
  - Per range bin, calculate the spatial covariance matrices in 32-bit floating point, perform matrix inversion, then store back to L3 memory.
  - Since the covariance is semi-positive definite, only upper triangle of the matrix is store.
- Range-azimuth heatmap calculation
  - See following slide for details.
  - Heatmap is stored back to L3 memory

Low Level Signal Processing Chain: Capon Beamforming (1): Algorithm Details

Let s(t) be the incoming waves after mixing to baseband, the sensor array signal to be processed is given by

$X(t) = A(\theta)s(t) + n(t)$

where $A(\theta) = (a(\theta_1), \ldots a(\theta_M))$ is the steering matrix
$a(\theta) = (e^{j2\pi y_1 \sin(\theta)}, \ldots, e^{j2\pi y_N \sin(\theta)})$ is the steering vector
M is number of angle bins
y_n is the sensor position normalized by wavelength

Capon BF approach is
$\theta_{capon} = \mathrm{argmin}_\theta\{\mathrm{trace}(A(\theta)*R_n^{-1}*A(\theta)^H\}$
where R_n is the spatial covariance matrix

Low Level Signal Processing Chain: Capon Beamforming (2) Implementation details

- Assumptions:
  - Receive antennas are equally spaced with distance $\lambda/2$

- Assuming slow motion scenario, Rn can be constructed using multiple chirps within a frame.

- Calculation of the solution can be significantly reduced with a(θ) constructed for equally spaced antenna, combining with the fact that Rn is an Hermitian matrix that is also persymmetric.

- Current implementation of the module consists of the following sub blocks:
  - Static clutter removal is option is added before Rn matrix generation by removing DC components per range bin. Input assumes 16-bit I/Q, and output is also in 16-bit I/Q format (trade precision for memory and cycles).
  - Per range bin, construct Rn using multiple chirps within a frame. Then Rn is inverted and the upper diagonal of the R_n^{-1} is stored in memory for each range bin.
  - Per range bin, calculate the Capon BF solution and store the angle spectrum in memory to construct the range-azimuth heatmap.
    - Since conventional BF using covariance matrix has very similar solution $\theta_{conventional} = \text{argmax}_{\theta}\{\text{trace}(A(\theta)*R_n*A(\theta)^H\}$, we have a fallback flag to use the same code to calculate conventional BF.
  - After detection, per detected point, estimate Doppler using the Capon beamweights and Doppler FFT.
  - Hardcoded for 4 and 8 antennas, and focused on 8-antenna test

- The module directly read from and write to L3 memory, with L1 data cache turned on for L3.

Low Level Signal Processing Chain: CFAR Detection

- 2-pass CFAR is perform on the range-azimuth heatmap.
- First pass is done per angle bin along the range domain, using CFAR-CASO
- Second pass in the angle domain is used to confirm the detection from the first pass, using CFAR-CASO as well
- CFAR module also access range-azimuth heatmap in L3 memory directly.

Low Level Signal Processing Chain: Doppler Estimation

- Doppler Estimation is done per detected point from range-azimuth CFAR detection
- Capon beamweights calculated from Capon beamforming for the particular range, and azimuth angle are used to filter the 1D FFT output for the corresponding range bin.
- FFTs will be performed on the filter outputs, peak search will be performed on the integrated signal, where peak index will be the Doppler index.
- Only one Doppler will be estimated per [range azimuth] detection pair.

Example Chirp Configuration

Input parameters	
Starting frequency (GHz)	77
Maximum range, Rmax (m)	6
Range resolution (cm)	4.9
Maximum velocity (km/h)	18.64
Velocity resolution (km/h)	0.297
Maximum velocity (m/s)	5.18
Velocity resolution (m/s)	0.08
Idle time (us)	30
ADC valid start time (us)	10
Excess ramping time (us)	1
Periodicity (ms)	50
Derived parameters	
Valid sweep bandwidth, B (MHz)	3061.22
Chirp time, Tc (us)	50.84
Ramp slope init (MHz/us)	60.22
Ramp slope parameter	1247
Ramp slope (MHz/us)	60
Maximum beat frequency (MHz)	2.41
Sampling frequency (Msps)	2.50
Number of samples per chirp	128
Total sweep bandwidth, Btotal (MHz)	3745.64
Idle time minimum (us)	7.00
Ramp end time	62.20
Carrier frequency, fc (GHz)	77.60
ADC valid start time, minimum (us)	5.5
Lambda (mm)	3.87
Chirp repetition period, max, Tr,max (us)	187.0
Chirp repetition period, Tr (us)	184.0
Nfft_range	128
Min number of chirp loops, Nchirp_loops	128
Nfft_doppler	128
Active frame time, Tframe_active (ms)	23.55
Range interbin resolution (cm)	4.90
Velocity interbin resolution (m/s)	0.08

Low Level Signal Processing Chain: Benchmarks for Example Chirp Config

- Range processing, in PING-PONG fashion
  - 14us out of available 92us ➔ loading = 15%
- Frame processing:
  - Available processing time: 50ms (frame period) – 23.5ms (active chirping time) = 26.5ms
  - Processing cost:
    - CaponBF: 3.3M cycles ➔ 5.5ms
    - CFAR: 180K cycles ➔ 0.3ms
    - Doppler estimation: 8k cycles per detected point. Assuming 100 detected points it comes with 800K cycles ➔ 1. 4ms
    - Total cost: 7.2ms ➔ loading = 27%

Low Level Signal Processing Chain: Memory Usage for Example Chirp Config

- L3 memory:
  - 600K used, out of 640K available.
- L2 memory:
  - 176K used, out of 256K available.
- L1P:
  - 4K configured as program cache
  - 28K configured as program RAM, out of which 21K used
- L1D:
  - 16K configured as data cache.
  - 16K configured as data RAM, all used as system scratch memory.