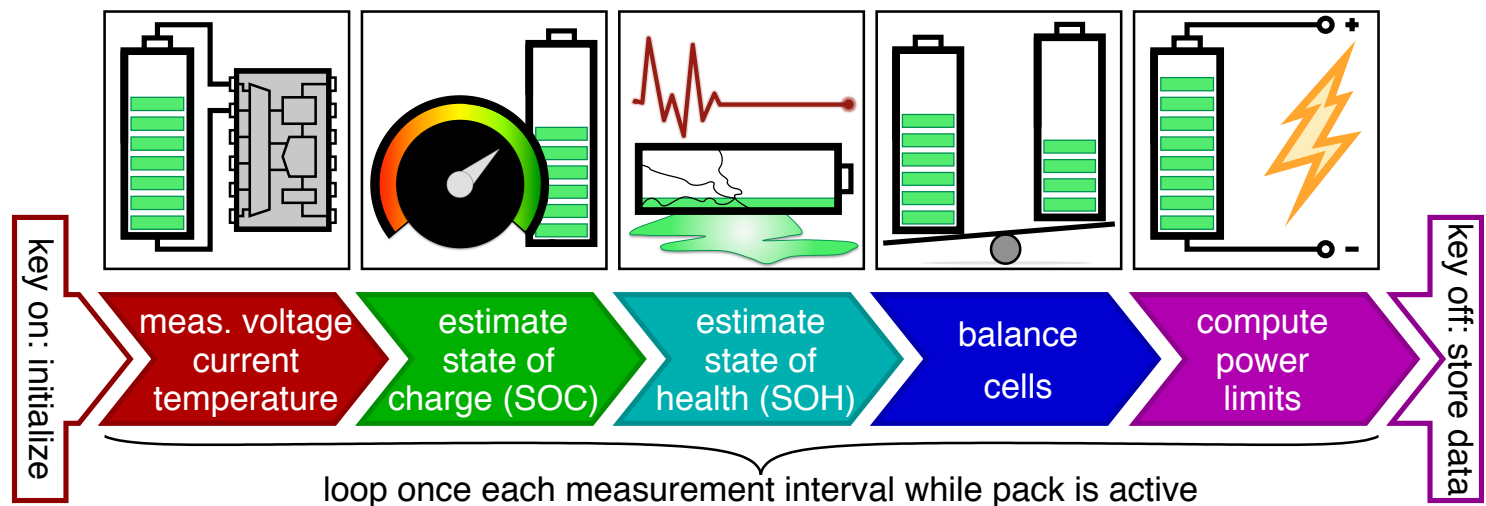


Cell Balancing

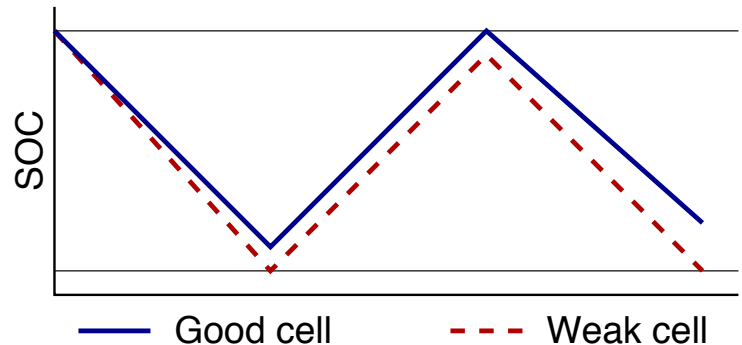
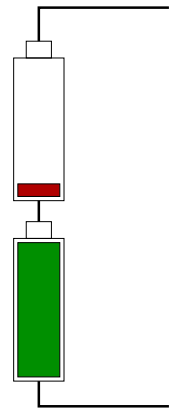
5.1: Causes (and not causes) of imbalance

- We've now explored the basic *estimation* tasks performed by a BMS.
- We now turn to the *control* tasks required by a BMS.
 - This chapter focuses on balancing or equalizing a battery pack.



- Balancing or equalizing is the process of modifying the level of charge in cells on a cell-by-cell basis.
- There are two basic approaches to balancing:
 - Passive balancing drains charge from cells having too much charge and dissipates drained energy as heat.
 - Active balancing moves charge from “high cells” to “low cells,” attempting to conserve energy in the battery pack.

- We will look at some balancing circuits later, but first we consider why balancing is important.
- Consider the trivial battery pack to the right.
 - Because the cells are out of balance, this pack can neither deliver nor accept energy/power.
- Generally, a cell that is “weak” in some sense will limit pack’s performance, and will ultimately render the pack useless unless cells are “balanced.”

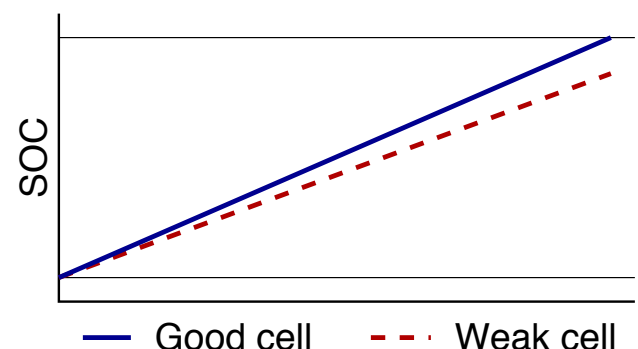


Causes of imbalance

- Imbalance is caused by anything that can make one cell’s SOC diverge from another.
- One example is when cells have different Coulombic efficiency:

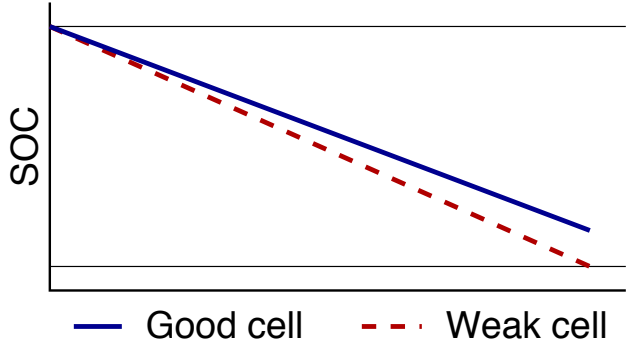
$$z(t) = z(0) - \frac{1}{Q} \int_0^t \eta(\tau) i_{\text{net}}(\tau) d\tau.$$

- Cells may start with same $z(0)$, have same capacity Q , and receive the same net current $i_{\text{net}}(t)$.
- But, because of different efficiency η , cell SOC’s diverge during charging.
- Imbalance can also be caused by cells having different net current from each other. That is, we need to carefully consider



$$i_{\text{net}}(t) = i_{\text{app}}(t) + i_{\text{self-discharge}}(t) + i_{\text{leakage}}(t),$$

where $i_{\text{app}}(t)$ is the battery-pack load current, $i_{\text{self-discharge}}(t)$ is the rate of cell self-discharge, and $i_{\text{leakage}}(t)$ is the current that powers attached BMS electronic circuitry.

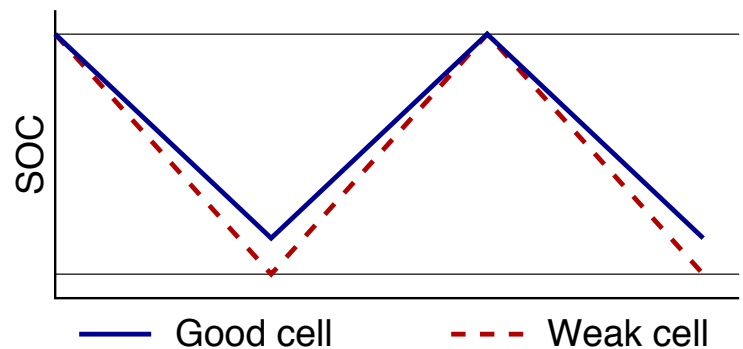
- Self-discharge rates of different cells can be different, leading to different $i_{\text{net}}(t)$.
 - Leakage current can be different for different cells, also leading to different $i_{\text{net}}(t)$.
- The bottom line is: when cells draw different net current, they become imbalanced.
- 
- And, since self-discharge rates, electronics' performance, and coulombic efficiency are functions of temperature, a pack temperature gradient can make the problem worse.
- Note that in all cases, it is *difference* in efficiency/ self-discharge/ leakage (etc.) that matters, not the absolute quantity thereof.
- If all cells are equally “bad”, there will be no increase in imbalance.

Not causes of imbalance

- Different cell capacities cause temporary imbalance that is automatically corrected when any cell returns to original SOC.

- For example: Remove 5 Ah from both a 6 Ah cell and a 5 Ah cell; then, replace the 5 Ah.

- SOC ends where they began.

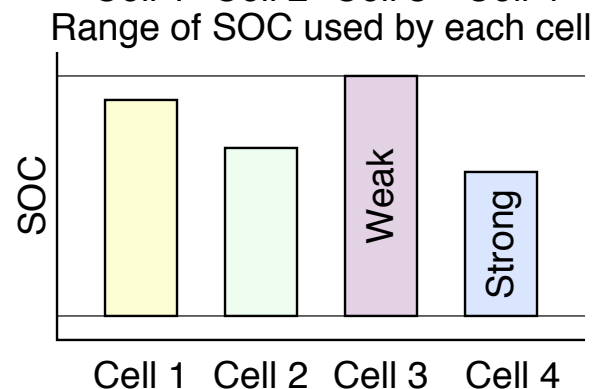
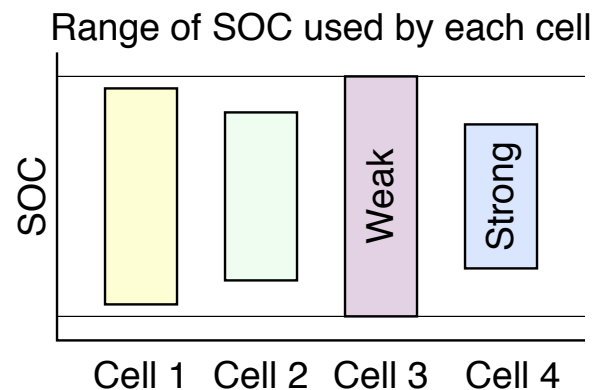
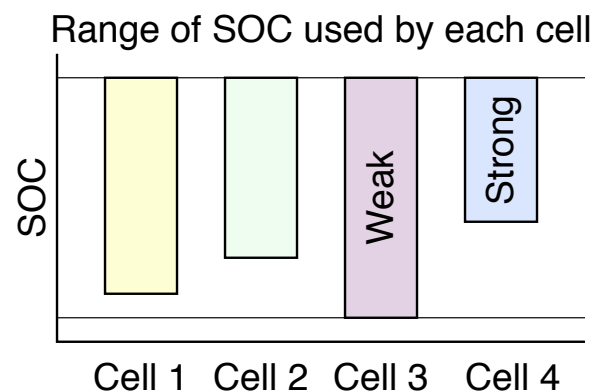


- But, different capacities do limit available pack energy:
 - Some energy stored in high-capacity cells is not available;
 - Fast active balancing can help.
- Similarly, different cell resistances cause cell voltages under load to be quite different, but not their SOC.
- A cell with high resistance will tend to hit an upper/ lower voltage limit before other cells, so will limit available pack power.
- Fast active balancing can (intentionally) bring pack to an out-of-balance condition to equalize the power that can be sourced/sunk from cells in the pack.

5.2: Design choices when implementing balancing

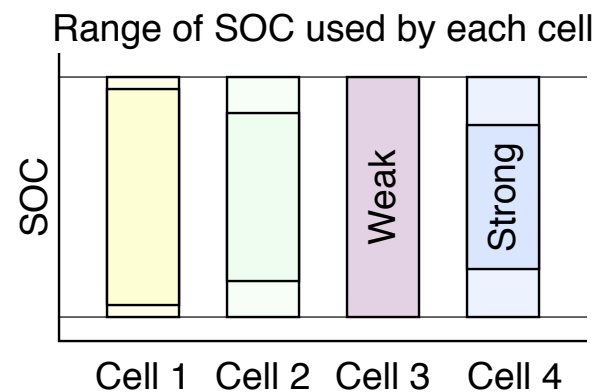
Design of balance setpoint

- A reasonable definition: “A balanced battery pack is one in which, at some point in its cycle, all the cells are at exactly the same SOC.”¹
 - Other definitions are possible where this condition is never met. However, this definition will be sufficient for now.
- But, we need to consider “what should that SOC be?”
- One choice is to balance at top SOC.
 - Maximizes energy that can be stored by the battery pack (good for EV).
 - But, some cell aging mechanisms accelerated at high SOC (bad).
- Or, could balance at a mid-SOC point.
 - Some impact on energy, but some applications don't care.
 - Maximizes pack's ability to accept or deliver power (good for HEV).
- Or, could balance at bottom SOC.
 - Somewhat lower energy than balancing at the top;
 - But, minimizes high-SOC-based aging on some cells;



¹ Andrea, D., Battery Management Systems for Large Lithium-Ion Battery Packs, Artech House, 2010, p. 23.

- However, the weak cells still experience the high-SOC-based aging, so we probably don't want to do this as it accelerates divergence between cells (weak cells age faster).
- If fast active balancing is used, the setpoint becomes dynamic, and the entire range of every cell can be used.
 - “Strong” cell experiences greater total load, so ages more quickly.
 - This is a self-regulating effect.



- These are *design choices*. There isn't a one-size-fits-all answer.
 - The controls engineer will need to consider the pros and cons corresponding to each alternative for every new application.

Design of real-time balance criterion

- Another question is, “what should be the real-time criterion for determining which cells to balance?”
 - Which balancer circuits should be “switched on” at any instant?
- We might choose to balance based on SOC estimates (until ΔSOC at the balance setpoint is small).
 - But, if SOC estimates are poor, can balance “wrong” cells.
- Could also choose to balance based on voltage measurements (until Δv small), especially if using fast active balancing.
 - Simpler, but voltage is a poor indicator of SOC.

- Wasteful since often balancing “wrong” cells.
- We could even balance based on total available energy (SOCs are never exactly equal at any point when balancing for this criterion).
 - Maximizes energy that can be extracted from battery pack before a design limit (usually on minimum cell voltage) is exceeded.
 - Can improve total available energy by moving charge from low-resistance cells to high-resistance cells to bolster their SOC (and hence their voltage).
 - Need an accurate cell model and state estimate for every cell.

Design of balance activity

- Another question we need to consider is, when to balance?

CONTINUOUSLY:

- Needed for HEV; needed if fast active balancing is used to maximize available pack energy and power.
- Harder to do “right” as SOC estimates must be accurate.

ON CHARGE ONLY:

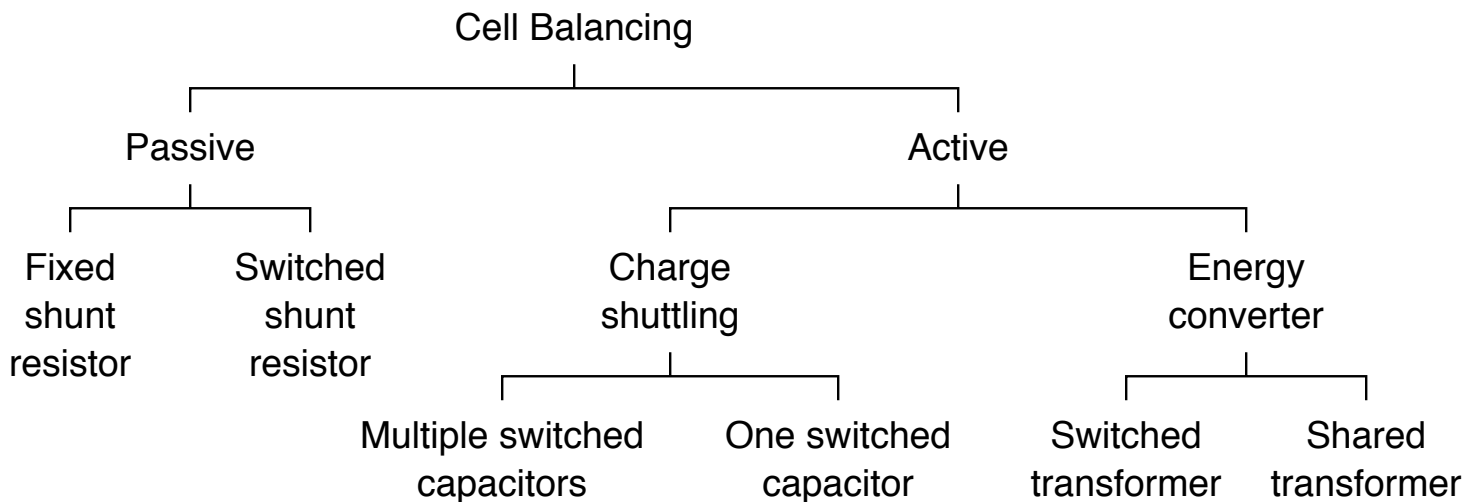
- Can be used for EV/PHEV/E-REV.
- Maximizes range: Dissipates charge only when plugged in.
- But, charging times must be larger to allow full balance.

PREDICTIVE:

- Predict where the end point will be on charge (etc).
- Proactively balance even when otherwise not obvious that balancing is needed.

5.3: Circuits for balancing (1): Passive

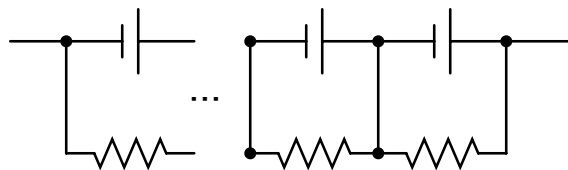
- There are a wide variety of generic electronics strategies that may be used in a cell-balancing system. The most common topologies are:



- We look at each of these in the following sub-sections.

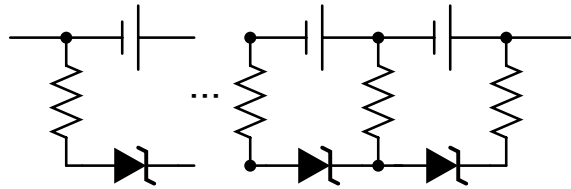
Passive: Fixed shunt resistor

- The simplest electronics designs are for passive balancing systems.
- The general idea is that a resistor is placed in parallel with each cell, and used to drain charge from that cell.
- The energy removed from the cell is dissipated as heat.
- The simplest design of all is the “fixed shunt resistor design.”



- The idea is that high-voltage cells will have greater balancing current, and so will self-discharge more quickly than low-voltage cells.
- However, note that the circuit is always dissipating charge, even when the pack is perfectly balanced.

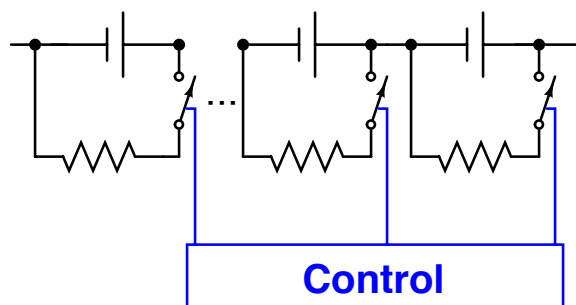
- A variation on the design uses zener diodes to “shut off” balancing when cell voltage drops below some point.



- The zener voltages are chosen to correspond to a “100 % SOC” setpoint. *e.g.*, about 2.2 V for a lead-acid cell.
- When a cell’s voltage is above the zener setpoint, the resistor path is activated, and that particular cell’s charge is depleted until the cell’s voltage drops below the zener setpoint.
- Note that this design works for chemistries where overcharge is tolerable, and the cell can “float.”
 - This includes lead-acid and nickel-based chemistries, but *not* lithium-ion chemistries.

Passive: Switched shunt resistor

- A variation on the above idea, which works for lithium-ion chemistries as well, is to replace the zener diode with a BMS-controlled switch.
 - This switch is some kind of transistor circuit.



- The electronics required to control the transistor make this design more complicated; however, it allows for much greater flexibility in balancing strategy.
- The BMS closes switches on cells having too much charge, allowing them to drain.
- Note that the added complexity is not as big an issue as it used to be.
 - Modern battery-stack monitoring chips have built-in circuitry to control either an internal transistor switch (for slow balancing) or an external transistor switch (for faster balancing).
- The primary advantage of any of these types of passive balancing is the simplicity (and hence, lower cost) of the circuitry involved, compared with active balancing designs.
- The drawbacks are:
 1. Energy is wasted as heat, which could be otherwise used productively.
 2. In a balance-at-top design, energy remains in cells when weak cell is completely discharged, which could be utilized by an active balancing system.
 3. Heat is generated. The power dissipated as heat is $P \approx V_{\text{nom}} \times I_{\text{balance}}$. For fast balancing, more heat is generated.
 - This generally imposes a high-wattage requirement on the balancing resistors, and a high-current rating on the balancing transistors.
 - The quantity of heat generated by balancing can be comparable to the heat generated by normal cell operation. This may also

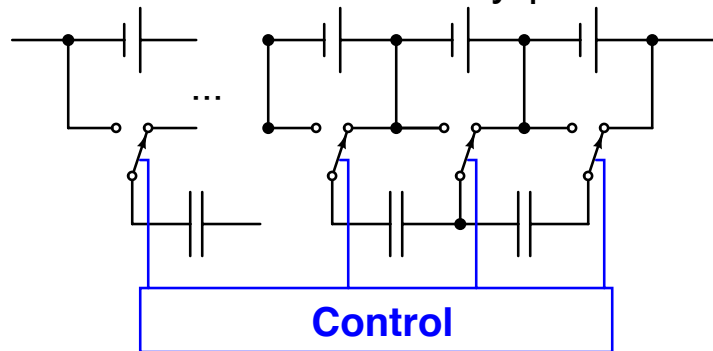
increase the cooling requirements for the battery-pack thermal management system, which is a significant expense.

4. Battery pack life could be shorter with respect to a pack with an active balancing design.
 - Pack life is determined by the weakest cell in the pack.
 - The active balancing can use strong cells to support weak cells, bringing the pack to a uniform end-of-life configuration.

5.4: Circuits for balancing (2): Active, capacitive

Active: Multiple switched capacitors

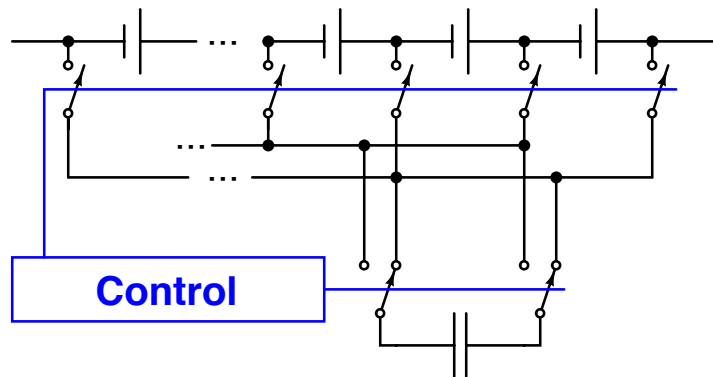
- Active balancing circuits break down into three general categories:
 - Move charge via switched capacitors;
 - Move energy via transformer/inductor designs;
 - Use dc-dc converter techniques to discharge high cells and charge low cells (we don't look at these in this course as the designs are quite expensive, with no clear benefit at this point).
- We look first at capacitor-based designs. In the circuit below, there is one fewer capacitor than there are battery-pack cells.



- The single-pole-double-throw (transistor circuit) switches repeatedly back and forth (no intelligence in the switching).
- Consider two neighboring cells. The higher-voltage cell charges the capacitor to its voltage, and then the lower-voltage cell discharges the capacitor to its voltage: charge moves to equalize cell voltages.
- Over the course of time, the entire battery pack can be equalized.
 - But, charge takes a long time to propagate from one end of the pack to the other.

Active: One switched capacitor

- An alternate design uses a single switched capacitor, with intelligent control:



- This allows direct movement of charge from a high-voltage to a low-voltage cell.
- A serious drawback of all capacitor-based designs is that they rely on a voltage difference between cells in order to work.
- Most lithium-ion chemistries have very little voltage variation between cells even if SOC varies a lot.
- The maximum energy that is transferred is $E = \frac{1}{2}C(v_{\text{high}}^2 - v_{\text{low}}^2)$.
- The energy can be related to a change in SOC: $E \approx (\Delta z)Qv_{\text{nom}}$.
- Suppose that $v_{\text{nom}} \approx \frac{v_{\text{high}} + v_{\text{low}}}{2}$. Then, equating the two energies gives

$$(\Delta z)Q \frac{v_{\text{high}} + v_{\text{low}}}{2} \approx \frac{1}{2}C(v_{\text{high}} + v_{\text{low}})(v_{\text{high}} - v_{\text{low}})$$

$$\Delta z \approx \frac{C}{Q}\Delta v,$$

where Q must be measured in coulombs for the units to work out.

EXAMPLE: Consider a 10 Ah cell (36 000 C). We would like to compute Δz when $\Delta v = 0.1$ V.

- We are free to select the capacitance value, but note that high-valued capacitors tend to have high resistance, so will charge slowly, a fact we have not taken into account in our simple approximations.
- Even if we select a (ridiculously) large value of $C = 1 \text{ F}$, we get

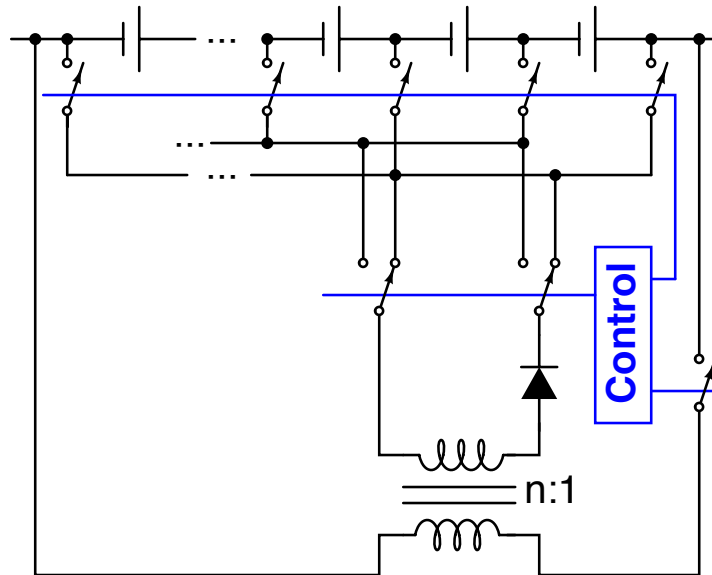
$$\begin{aligned}\Delta z &\approx \frac{1}{36\,000} \times 0.1 \\ &\approx 3 \times 10^{-6}.\end{aligned}$$

- At this rate, it will take forever to equalize!
- Capacitor-based designs make most sense for EV, where voltage variation can be fairly large.
- They don't make much sense for HEV, as cells are operated in smaller SOC window, and voltage variation tends to be very small.

5.5: Circuits for balancing (3): Active, inductive and dc-dc

Active: Switched transformer

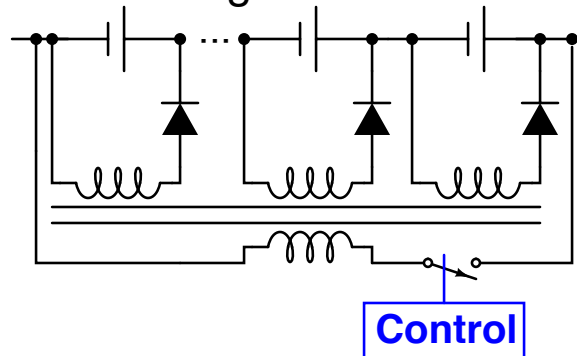
- An alternative approach, which can move a great deal of charge at once, is to use a transformer.



- Rapidly switching the primary creates an approximate AC waveform, reproduced at the secondary.
 - Primary is connected across n cells;
 - Transformer is wound with a $n : 1$ ratio.
 - Output of transformer is decreased in voltage by factor of n , but increased in current by factor of n .
- The diode plus switches select into which cell to dump the charge.
- Much more efficient than passive balancing; much faster than capacitive methods; but also expensive due to transformer and electronics costs.
- Presently, silicon vendors are working to create automated controls chips that will make this design much more feasible.

Active: Shared transformer

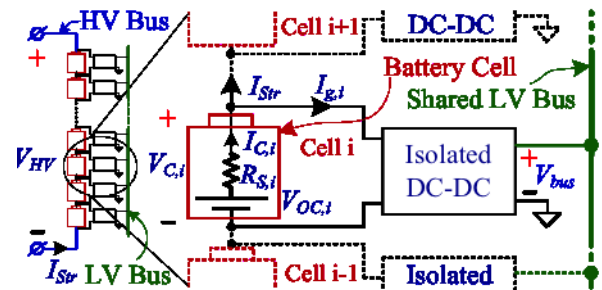
- Finally, a simplified variation of the prior scheme is to use a transformer with custom winding and a diode circuit.



- The control rapidly switches the primary; diodes route the current. Balancing is automatic without sophisticated algorithms.

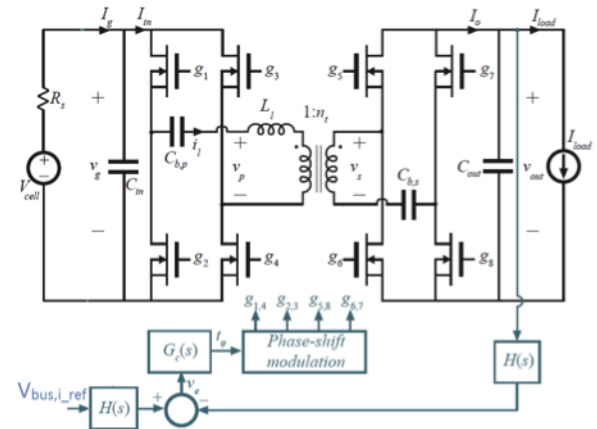
Active: Shared bus

- The primary problem with active balancing is its cost.
- A new approach uses one small dc–dc converter per cell and a capacitive shared low-voltage bus to perform balancing.²
- A balancing metric is mapped to a dc level between about 9 V and 14 V.
 - Metric might be cell SOC, voltage, or something more advanced.
- For example, metric might be designed to promote differential power processing, relatively increasing stress on “strong” vs. “weak” cells.
 - Enhances life, and brings entire pack to homogeneous end-of-life.



² M.M. Ur Rehman, M. Evzelman, K. Hathaway, R. Zane, G.L. Plett, K. Smith, E. Wood, and D. Maksimovic, “Modular Approach for Continuous Cell-level Balancing to Improve Performance of Large Battery Packs,” ECCE 2014.

- Principle of operation: Using controlled dc–dc converter,
 - Transfer charge from low-voltage shared bus to cell if this cell's metric is below the shared-bus voltage;
 - Transfer charge from cell to low-voltage shared bus if this cell's metric is above the shared-bus voltage.
- Can power auxiliary loads from the low-voltage bus: if load power is large and consistent enough, can replace bidirectional dc–dc converters with unidirectional converters.
 - (H-bridge dc–dc converter depicted to right.)
- Costs are reduced by processing only the small mismatch power between battery cells and the simplicity of the modular structure with a single isolated shared bus with no additional communications.
- The overall solution can be better than cost-neutral (vs. passive balancing) because it can both (1) balance, and (2) replace the costly large dc–dc converter presently used to convert pack voltage to 12 vdc for vehicle accessories.



5.6: How quickly must I balance a pack?

- Must balance at least as quickly as the pack becomes unbalanced, for long-term balancing needs.
- Battery pack simulation is an excellent tool to evaluate how quickly pack can reach imbalance.
 - Select random cell characteristics typical of variation expected in real cells;
 - Simulate battery pack over many repeated realistic drive scenarios.
 - Gather statistics on how quickly cells become imbalanced.
 - Then, simulate balancing: see how quickly can balance.
- In this section, we examine code designed to simulate packs having random cell characteristics
 - We'll consider the main routine `simRandPack.m` in sections.
- The first section gives help information, sets up random parameters, sets up simulation variables.
 - A battery pack comprising N_S cells is created, and simulated for N_C discharge/charge cycles.
 - Each discharge cycle comprises repetitions of the power versus time profile in `cycleFile`.
 - The ESC-format cell model is passed as `model`.
 - The `randOptions` fields are set to “0” for a standard simulation; “1” for random values, as unpacked below the function header.

```

% -----
% simRandPack: Simulate battery pack having Ns cells in series for Nc
% discharge/charge cycles, where all cells in pack can have random
% parameter values (e.g., capacity, resistance, etc.)
%
% Assumes no hysteresis in the cell model (this could be changed
% fairly easily; hysteresis makes results more difficult to interpret,
% so this assumption is okay for a first analysis, at least).
% -----
function packData = simRandPack(Ns,Nc,cycleFile,model,randOptions)

tOpt = randOptions(1); qOpt = randOptions(2); rOpt = randOptions(3);
sdOpt = randOptions(4); cOpt = randOptions(5); lOpt = randOptions(6);
profile = load(cycleFile); % e.g., 'uddsPower.txt'

% -----
% Create storage for all cell states after completion of each cycle
% -----
packData.storez = zeros([Ns Nc]); % create storage for final SOC
packData.storeirc = zeros([Ns Nc]);

% -----
% Initialize default states for ESC cell model
% -----
maxSOC = 0.95; % cell SOC when pack is "fully charged"
minSOC = 0.1; % cell SOC when pack is "fully discharged"
z = maxSOC*ones(Ns,1); % start fully charged
irc = zeros(Ns,1); % at rest
ik = zeros([Ns 1]); % current experienced by each cell

```

■ The next section populates random variables for this battery pack

```

% Set cell temperatures based on tOpt
if tOpt, % set to "if 1," to execute, or "if 0," to skip this code
    T = 22.5 + 5*rand([Ns 1]);
else
    T = 25*ones([Ns 1]);
end
% Set self-discharge "cell temperature"
Tsd = T - 5 + 10*rand([Ns 1]);

% Set cell module leakage current based on lOpt

```

```

if lOpt,
    leak = 0.01 + 0.002*rand([Ns 1]);
else
    leak = 0.01*ones([Ns 1]);
end

% -----
% Default initialization for cells within the pack
% Note that since T has Ns elements, there is one parameter value
% per cell (even if all turn out to be identical)
% -----

q    = getParamESC('QParam',T,model);
rc   = exp(-1./abs(getParamESC('RCPParam',T,model)));
r    = (getParamESC('RParam',T,model)).*(1-rc);
r0   = getParamESC('R0Param',T,model);
rt   = 2*0.000125; % 125 microOhm resistance for each tab
maxVlim = OCVfromSOCtemp(maxSOC,T,model);
minVlim = OCVfromSOCtemp(minSOC,T,model);
eta  = ones([Ns 1]);

% -----
% Modified initialization for cell variability
% -----
% Set individual random cell-capacity values
if qOpt, % set to "if 1," to execute, or "if 0," to skip this code
    q=q-0.25+0.5*rand([Ns 1]); % random capacity for ea. cell
end

% Set individual random cell-resistance values
if rOpt, % set to "if 1," to execute, or "if 0," to skip this code
    r0 = r0-0.0005+0.0015*rand(Ns,1);
end
r0 = r0 + rt; % add tab resistance to cell resistance
R = sum(r0,1);

% Set individual random cell-coulombic-efficiency values
if cOpt, % set to "if 1," to execute, or "if 0," to skip this code
    eta = eta - 0.001 - 0.002*rand([Ns 1]);
end

```

- Now, simulate the battery pack, store results.

```

% -----

```

```

% Now, simulate pack performance using ESC cell model.
% -----
theCycle = 1; theState = 'discharge';
disCnt = 0; % start at beginning of profile
fprintf(' Cycle = 1, discharging... ');
while theCycle <= Nc,
    v = OCVfromSOCtemp(z,T,model); % get OCV for each cell
    v = v - r.*irc; % add in capacitor voltages
    V = sum(v); % Total voltage excluding I*R
    vt = v-ik.*r0; % Cell terminal voltages

    switch( theState )
        case 'discharge';
            % Get instantaneous demanded pack power, repeating profile
            P = profile(rem(disCnt,length(profile))+1);
            % Compute demanded pack current based on unloaded voltage
            I = V/(2*R) - sqrt(V^2/R^2 - 4*P/R)/2;
            % Default cell current = pack current
            ik = I*ones(Ns,1);
            if I < 0, % If we happen to be charging this moment
                ik = ik.*eta;
            end
            if min(z) <= minSOC || min(vt) < minVlim, % stop discharging
                theState = 'charge';
                chargeFactor = 1;
                ik = 0*ik;
                fprintf('charging... ');
            end
            disCnt = disCnt + 1;

        case 'charge';
            % start charging @ 6.6kW, then taper
            P = -6600/chargeFactor;
            I = V/(2*R) - sqrt(V^2/R^2 - 4*P/R)/2;
            I = max(-min(q),I); % limit to 1C charge rate max
            ik = I*eta; % Charge coulombic eff.
            if max(vt)>=maxVlim,
                if chargeFactor > 32, % bail after 6.6kW/32 charge
                    packData.storez(:,theCycle) = z;
                    packData.storeirc(:,theCycle) = irc;
                    theState = 'discharge';
                    disCnt = 0;
                end
            end
    end
end

```

```

        ik = 0*ik;
        theCycle = theCycle + 1;
        if theCycle <= Nc,
            fprintf('\n Cycle = %d, discharging... ',theCycle);
        end
    end
    chargeFactor = chargeFactor*2;
end
otherwise
    error('charge/discharge state has been corrupted')
end

% Simulate self discharge via variable resistor in parallel
if sdOpt == 1,
    rsd = ((-20+0.4*Tsd).*z + (35-0.5*Tsd))*1e3;
    ik = ik + vt./rsd;
end

% Simulate leakage current
ik = ik + leak;

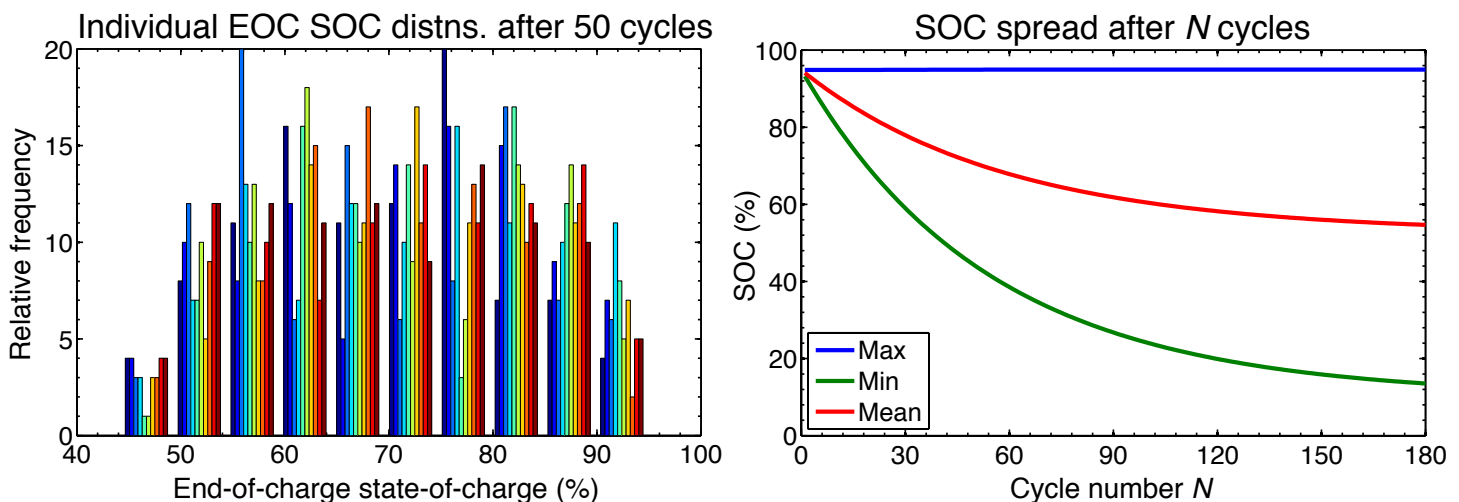
z = z - (1/3600)*ik./q; % Update each cell SOC
irc = rc.*irc + (1-rc).*ik; % Update resistor currents
end % end while
fprintf('\n');
packData.q = q; packData.rc = rc; packData.eta = eta;
packData.r = r; packData.r0 = r0; packData.Tsd = Tsd;
packData.T = T; packData.leak = leak;
end

```

- This code would generally be called by a “wrapper function” that simulates many random packs (to gather statistics).
- Any and all combination(s) of random options can be chosen, to explore sensitivity of imbalance to each effect.
- the `packData` structures can be stored for later analysis using balancing algorithms, to see how quickly balancing must occur.

5.7: Some results of balancing simulations

- We run the given code for 100 random battery packs, each having 100 cells, for 180 discharge/charge profiles for each of a number of scenarios.
- Histograms of end-of-charge SOC showing data from all 100 random packs gives an idea of the variation expected without balancing after some number of discharge/charge cycles.
- Plots of minimum/maximum/mean SOC versus cycle give an idea of how quickly SOC can diverge.
- The worst-case scenario attempted shows that imbalance can reach 50 % after only 50 days' driving, for reasonable assumed parameters for good cells.



- Divergence slows down simply because the unbalanced pack cannot be run as long during the “discharge” portion of the cycle. There is much less available energy.
- So, for this example, we must be able to balance at least 1 % per drive cycle per cell.

- For a 7.7 Ah pack, this is at least 77 mAh, per cell, per balancing period (will want to over-design capability).
 - Passive balancing can be fast enough to keep pack in balance.
 - Passive balancing is not fast enough to maximize energy/power or to extend life via quick charge transfer.
- Also, if vehicle is stored for extended periods, more balancing may be necessary due to self discharge.

Where from here?

- We have now discussed all the major estimation tasks, with the exception of power-limit estimation.
- We devote the remainder of the course to topics surrounding power limits.